

Méthodes de détection des similarités structurales : caractérisation des motifs conservés dans les familles de structures pour l'annotation des génomes

THÈSE

présentée et soutenue publiquement le 9 décembre 2005

pour l'obtention du

Doctorat de l'Université Pierre et Marie Curie

par

Mathilde Carpentier

Composition du jury

Rapporteurs : Dr. Marie-France Sagot
Dr. Jean-François Gibrat

Examinateurs : Pr. Catherine Etchebest
Pr. Jean Cognet

Directeurs de thèse : Dr. Joël Pothier
Pr. Pierre Netter

Remerciements

Je pourrais écrire à la va vite des remerciements mais je trouve que les personnes qui m'ont entourées méritent bien mieux. J'espère que les rapporteurs, Marie-France Sagot et Jean-François Gibrat m'excuseront de préférer d'attendre un peu. Je les remercie de tout cœur d'accepter de relire cette thèse.

Table des matières

Introduction générale	1
I État des connaissances	5
1 Les Protéines	7
1.1 Les protéines	8
1.2 La structure des protéines	10
1.2.1 Détermination	10
1.2.2 Description géométrique du squelette	14
1.2.3 Autres propriétés géométriques	16
1.3 La prédiction de la structure des protéines	17
1.3.1 La modélisation par « homologie »	17
1.3.2 La reconnaissance de repliements	17
1.3.3 La modélisation <i>ab initio / de novo</i>	18
1.4 L'annotation structurale	18
2 Comparaison de deux structures	21
2.1 Introduction	22
2.1.1 Les premières comparaisons de structures	22
2.1.2 Définitions	23
2.1.3 Objectifs de ces méthodes	23
2.2 Description au niveau atomique	26
2.2.1 Graphes et comparaison structurale	26
2.2.2 Construction de motifs structuraux	28

2.2.3	Hachage géométrique	30
2.2.4	Méthodes fondées sur la forme	31
2.3	Description au niveau du squelette peptidique	32
2.3.1	Les mesures de similarité : les RMSD	32
2.3.2	Les coordonnées cartésiennes des C_α	33
2.3.3	Les coordonnées internes : les distances « internes »	42
2.3.4	Les repères internes	48
2.3.5	Les coordonnées internes : les angles	49
2.3.6	Méthodes utilisant plusieurs descriptions	51
2.4	Description en éléments de structures secondaires	54
2.4.1	Méthodes avec les SSE représentés par des vecteurs	54
2.4.2	Autres méthodes de comparaison	56
2.4.3	Méthodes avec SSE représentés par d'autres caractéristiques	60
2.4.4	Alphabets structuraux	64
3	Comparaison multiple de structures	67
3.1	Enrichissement structural de motifs de séquences	68
3.2	Méthodes avec comparaison des paires de structures	69
3.2.1	Méthodes de superposition multiple avec correspondance donnée	69
3.2.2	Méthodes d'alignement hiérarchique et méthodes de structures moyennes	70
3.2.3	Méthodes avec structure pivot ou moyenne	73
3.3	Méthodes sans alignement des paires	75
3.3.1	Méthodes utilisant les graphes	75
3.3.2	Méthode de hachage géométrique	76
3.3.3	Méthodes de recherche de motifs répétés	76
4	Comparaison des méthodes décrites	79
4.1	La qualité des alignements	80
4.1.1	Comparaison manuelle d'alignement	80
4.1.2	Validation de la qualité d'un alignement par le calcul d'un score.	81
4.2	La sensibilité des méthodes	83
4.3	Conclusion	84
5	Les classifications existantes	87
5.1	Classifications hiérarchiques	88
5.1.1	SCOP	88
5.1.2	CATH	89

5.2	Classifications non hiérarchiques	90
5.2.1	FSSP	90
5.2.2	DALI Domain Dictionary	91
5.2.3	CE database	91
5.3	Alignements multiples de familles de structures	92
5.3.1	3D-PSSM	92
5.3.2	HOMSTRAD	93
5.3.3	PALI	93
5.3.4	LPFC	93
5.4	Banques d'alignements structuraux	93
5.4.1	HOMALDB	94
5.4.2	MMDB	94
II	Nouvelles méthodes	95
6	Yakusa : une nouvelle méthode de recherche de similarités structurales	97
6.1	Méthode	98
6.1.1	Description de la structure tridimensionnelle des protéines	98
6.1.2	Description générale de l'algorithme	99
6.1.3	L'automate	100
6.1.4	Recherche, sélection et extension des graines	108
6.1.5	Sélection des SHSP	110
6.1.6	Calcul du score final et classement	110
6.1.7	Complexité - Temps d'exécution	113
6.1.8	Implémentation	114
6.2	Tests et résultats	115
6.2.1	Test général	115
6.2.2	LiveBench	117
6.2.3	Sensibilité aux petites variations de structures	120
6.2.4	Cas des protéines multi-domaines	120
6.2.5	Cas non triviaux	121
6.2.6	Détection des nouveaux replis	122
6.3	Serveur	123
6.4	Conclusion	123

7 Comparaison de multiple de structures	125
7.1 Méthode des <i>m</i> -diagonales	126
7.1.1 Méthode	126
7.1.2 Adaptation aux structures	126
7.1.3 Algorithme	127
7.1.4 Paramètres et temps d'exécution	129
7.1.5 Exemples	129
7.1.6 Conclusion	130
7.2 Méthode du <i>Gibbs Sampling</i> appliquée aux structures	133
7.2.1 Méthode générale	133
7.2.2 Temps d'exécution	136
7.2.3 Exemples	136
7.2.4 Conclusion	137
7.3 Recherche de motifs répétés	138
7.3.1 Présentation générale	138
7.3.2 Quelques définitions	139
7.3.3 Construction des motifs	140
7.3.4 L'algorithme : implémentation avec des piles	142
7.3.5 Applications aux structures protéiques	144
7.3.6 Quelques exemples sur les cytochromes P450	146
7.3.7 Conclusions	146
III Résultats	149
8 Constitution des familles structurales	151
8.1 Classification des paires de structures	153
8.1.1 Présentation des deux méthodes de classification	153
8.1.2 Jeu de données d'apprentissage	156
8.1.3 Résultats des classifications	157
8.2 Classification en familles des structures	162
8.2.1 Présentation de MCL	162
8.2.2 Sensibilité aux paramètres et à la longueur des protéines	164
8.2.3 Quelques exemples de familles structurales	165
8.3 Constitution des « coeurs »	167
8.4 Conclusion et perspectives	168

IV Conclusion et perspectives	169
Bibliographie, annexes et index	175
A Algorithmes connus utilisés en bioinformatique	
A.1 La programmation dynamique NWS (Needleman and Wunsch, 1970)	197
A.2 <i>p-value, e-value</i> et distribution des valeurs extrêmes	198
A.3 Algorithmes de Triades	198
B Formules chimiques des vingt acides aminés	205
C Publications – conférences	209

Abréviations et acronymes

AFP :*Aligned Fragment Pair*

ADN : Acide DésoxyriboNucléique

ARN : Acide RiboNucléique

CATH : *Class Architecture Topology Homology*

DDD : *Dali Domain Dictionary*

FSSP : *Fold classification based on Structure-Structure alignment of Proteins*

MMDB : *Molecular Modeling DataBase*

PDB : *Protein Data Bank*

RMN : Résonance Magnétique Nucléaire

SSE : *Structural Secondary Element*

SHSP : *Structural High Scoring Pair*

SCOP : *Structural Classification Of Proteins*

Introduction générale

La compréhension du fonctionnement des organismes passe par la connaissance et la compréhension des rôles et des interactions des molécules les constituant. Les molécules biologiques peuvent être classées en quatre catégories principales : les acides nucléiques (ADN et ARN), les acides gras ou lipides, les sucres ou glucides et les protéines. Les protéines sont responsables de la plupart des fonctions cellulaires ; elles interviennent à tous les niveaux, de la duplication de l'ADN au métabolisme chimique, en passant par la structuration de la cellule et la transmission de signaux. Malgré leurs activités très variées, ces molécules sont assez homogènes : elles sont toutes composées d'une suite d'acides aminés liés par une liaison covalente formant la liaison peptidique. Il n'existe que 20 acides aminés. Cette suite d'acides aminés est appelée séquence protéique et peut être déterminée par différentes méthodes de séquençage.

Les protéines adoptent une organisation spatiale déterminée. Elle est cruciale pour assurer leur fonction. La structure des protéines est de nos jours déterminée principalement par deux méthodes : la diffraction aux rayons X, nécessitant une cristallisation de la protéine et la Résonance Magnétique Nucléaire (RMN). Le nombre de structures protéiques résolues croît rapidement d'année en année (30000 molécules sont recensées dans la *Protein Data Bank* en mai 2005) (Bernstein et al., 1977; Berman et al., 2000). Ce nombre important de structures permet d'aborder trois grandes questions : quelles sont les relations entre la séquence et la structure ? La structure détermine-t-elle la fonction ? Comment évoluent les structures ? Mon travail se situe dans cette perspective et j'ai œuvré principalement dans le développement de méthodes de comparaisons structurales systématiques ou multiples.

De nos jours, beaucoup de séquences biologiques sont connues, les méthodes de séquençage étant de plus en plus efficaces. Il est par contre long et souvent difficile de déterminer la structure d'une protéine par diffraction aux rayons X ou RMN. Beaucoup moins de structures protéiques sont donc connues. Il existe une alternative : les méthodes de prédition de la structure à partir de la séquence protéique. Ces méthodes reposent sur l'hypothèse que toute l'information nécessaire au repliement d'une protéine est inhérente à l'ordre de ses acides aminés, *i.e.* la séquence protéique, hypothèse reposant sur les travaux de Christian Anfinsen. Parmi ces méthodes, celles de « reconnaissance de repliements. » (*threading*), consistent à mesurer la « compatibilité » d'une séquence de structure inconnue avec tous les repliements protéiques connus. Pour chaque repliement (« cœur »), un grand nombre d'« alignements » de la séquence sur ce repliement « patron » sont effectués. Puis, une pseudo « énergie » est calculée pour chacun de ces alignements, sur la base de potentiels statistiques pré-déterminés à partir de toutes les structures de la PDB. Il est possible d'inférer - le cas échéant - que la séquence a une structure similaire à celle de ce cœur structural qui a fourni la meilleure énergie (un calcul statistique peut être effectué pour établir la vraisemblance de ce repliement).

Ces techniques peuvent fournir des modèles 3D de protéines à « retravailler » en modélisation moléculaire, mais surtout, elles sont de plus en plus utilisées à des fins d'annotation de séquences inconnues. Elles permettent en effet de repérer des similarités indécelables par les méthodes utilisant uniquement l'information de séquence. Comme ce type d'annotation prend en compte des données provenant de la structure des protéines, le terme d'annotation « structurale » est utilisé. Cet aspect

prend toute son importance aujourd’hui où l’on constate qu’environ la moitié des protéines des nouveaux génomes séquencés sont « orphelines », c’est-à-dire ne ressemblent en séquences à aucune protéine connue à ce jour. D’autre part, il est désormais admis que le nombre de structures 3D protéiques possibles est limité à quelques milliers et que ce nombre est de plusieurs ordres de grandeur en dessous du nombre de séquences possibles.

Ces méthodes poussent à étudier de plus en plus loin les relations entre les séquences et les structures. Il est remarquable d’observer combien de séquences protéiques différentes peuvent pourtant avoir le même type de repliement. Cette « dégénérescence séquence-structure », démontrée dans le cas des structures connues, et le nombre limité de structures possibles sont en partie confirmés par le faible nombre de nouveaux repliements trouvés actuellement par diffraction aux rayons X.

Généralement, ces repliements similaires comportent plusieurs blocs structuraux conservés organisés de manière similaire dans l’espace. Pour déterminer ces blocs structuraux, il faut d’abord classer les structures protéiques, *i.e.* regrouper les structures ayant un repliement similaire. Il est ensuite possible de chercher à déterminer les blocs structuraux conservés parmi les structures d’un même groupe (ou famille). La classification des structures est effectuée par comparaison de toutes les paires de structures tandis que lors de l’identification des blocs structuraux conservés toutes les structures d’une même famille sont comparées. Deux types d’outil de comparaison sont donc nécessaires, l’un permettant de comparer deux structures et l’autre permettant de comparer plusieurs structures simultanément. Il faut aussi établir une procédure de classification à partir des similarités structurales trouvées.

L’objet de ce travail a été d’étudier la conservation structurale locale parmi les protéines dont la structure est connue. Ainsi, les familles de familles de structures ont été construites d’après les blocs structuraux déterminés entre les paires de structures. Beaucoup de méthodes de recherche des similarités structurales ont été développées car différents types d’information peuvent être pris en compte et la comparaison étant un problème NP complet, beaucoup d’algorithmes ont été développés. Cependant, il n’existait pas de méthode permettant une recherche rapide et locale des similarités structurales. La première méthode de comparaison structurale mise en place, YAKUSA (Carpentier et al., 2005) atteint ces deux objectifs. Elle permet de trouver les similarités structurales entre une structure dite requête et toutes les structures d’une banque. Elle est basée sur une description simple de la chaîne polypeptidique (les angles α) et est inspirée de l’algorithme de Aho-Corasick (Aho and Corasick, 1975) qui permet de rechercher en parallèle plusieurs motifs dans un texte.

Sur la base des similarités structurales locales, les structures peuvent ensuite être regroupées en familles. Cependant, plusieurs informations peuvent être prises en compte pour établir la similarité entre deux structures comme le nombre de blocs structuraux, le nombre de résidus dans ces blocs... Ces différentes informations ont été prises en compte en essayant de classer les paires de structures en paires de structures similaires et non-similaires. Cette classification en deux classes a été effectuée en calculant pour les différentes informations deux distributions Normales Multivariées, une pour chaque classe. Ensuite, un graphe a été construit, un sommet représentant une structure et une arête étant définie si deux structures ont été classées précédemment comme similaires. Les familles structurales

dans ce graphe sont constituées des protéines partageant le plus d'arêtes. Une méthode de recherche des sous-graphes fortement connexes a donc permis d'identifier les familles de structures similaires (méthode MCL (van Dongen, 2000; Enright et al., 2002).

Pour déterminer les blocs structuraux communs aux protéines d'une même famille, trois méthodes ont été développées, chacune répondant à un problème spécifique. La première méthode permet de déterminer les blocs structuraux conservés à partir des blocs structuraux définis entre les paires de structures (Carpentier et al., 2004). Elle utilise la même description des structures que YAKUSA. Cependant, comme les blocs structuraux communs à plusieurs structures ne peuvent pas toujours être construits à partir des blocs structuraux des paires de structures, une seconde méthode a été développée (Carpentier et al., 2004). Elle permet donc de déterminer les blocs structuraux sans comparaison des paires de structures. La troisième méthode utilise une description des structures plus fine que les deux autres. C'est une méthode plus lourde mais plus précise et exhaustive (Pisanti et al., 2005a; Pisanti et al., 2005c; Pisanti et al., 2005b).

Dans la première partie, le premier chapitre est une brève présentation des protéines. Dans le second chapitre les méthodes développées à ce jour pour comparer deux structures protéiques sont abordées et dans le troisième sont présentées les méthodes permettant de comparer simultanément plusieurs structures. Les principales classifications des structures sont exposées dans le quatrième chapitre. Pour conclure cette partie, quelques comparaisons des résultats obtenus par différentes méthodes sont expliquées.

Les méthodes de comparaison structurale développées lors de ce travail sont de deux catégories, un chapitre est consacré à chacune dans la seconde partie.

Les outils de comparaison structurale étant mis en place, la procédure de classification ainsi que les résultats préliminaires obtenus sont présentés dans la troisième partie. Le dernier chapitre est une présentation de la procédure de classification des structures protéiques.

Première partie

État des connaissances

CHAPITRE 1

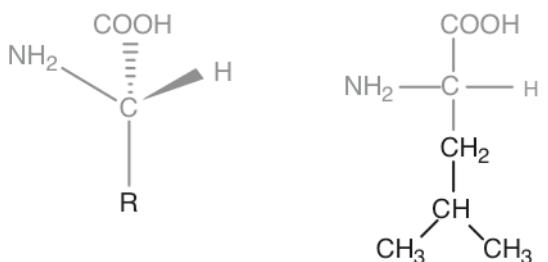
Les protéines : structures et fonctions

1.1 Les protéines

Les protéines sont parmi les principaux types de molécules du vivant (les autres étant les acides nucléiques, les lipides et les glucides). Dans la cellule, elles assurent la grande majorité des fonctions enzymatiques, et une bonne part des fonctions de maintien de la structure et de transport.

1) Composition des protéines - Les acides aminés

Les protéines¹ sont des polymères composés de 20 unités de base, les acides aminés (ou résidus). Ces derniers sont composés d'un atome carbone central (C_α) lié à un groupe aminé (NH_2), à un groupe carboxylique ($COOH$), à un atome d'hydrogène (H_α), et à un des 20 groupements chimiques différents appelés chaînes latérales (figure 1.1 et annexes figures B.1 et B.2). Les résidus et leurs fonctions chimiques différentes permettent la diversité fonctionnelle des protéines. Dans la chaîne polypeptidique, le groupe α -carboxylique d'un acide aminé est lié au groupe α -aminé de l'amino-acide suivant par une liaison amide (liaison peptidique $-CO-NH-$, voir figure 1.2). Les protéines sont des chaînes polypeptidiques fonctionnelles, et la plupart des protéines naturelles contiennent entre 50 et 2000 résidus d'acide aminé (Stryer, 1994). La chaîne non ramifiée des résidus a une direction, elle commence à l'extrémité aminée (extrémité NH_2 terminale) et se termine à l'extrémité carboxy terminale. La chaîne d'atomes répétant régulièrement les liaisons peptidiques se nomme le squelette peptidique (ou carboné si seuls les C_α sont pris en compte). La liaison peptidique est rigide et plane à cause du caractère partiel de double liaison de la liaison $-CO-NH-$, mais des rotations sont possibles autour des autres liaisons ($C_\alpha - CO$ et $NH - C_\alpha$).



Leucine (Leu, L)

FIG. 1.1 – Gauche : structure générale d'un acide aminé. Droite : Formule chimique de la leucine.

Les interactions entre atomes des chaînes latérales et celles entre atomes du squelette polypeptidique sont responsables de la structure des protéines. Trois types de liaisons non covalentes dans les protéines sont observées : les liaisons hydrogène, les liaisons de Van der Waals, et les liaisons électrostatiques. La liaison hydrogène est formée lorsqu'un atome d'hydrogène est partagé entre un donneur d'hydrogène (l'atome lié covallement à l'hydrogène) et un accepteur d'hydrogène. Les liaisons électrostatiques interviennent entre atomes chargés positivement et négativement. Les liaisons

¹ ou peptides qui sont une chaîne comportant moins de cinquantaine acides aminés reliés par des liaisons peptidiques.

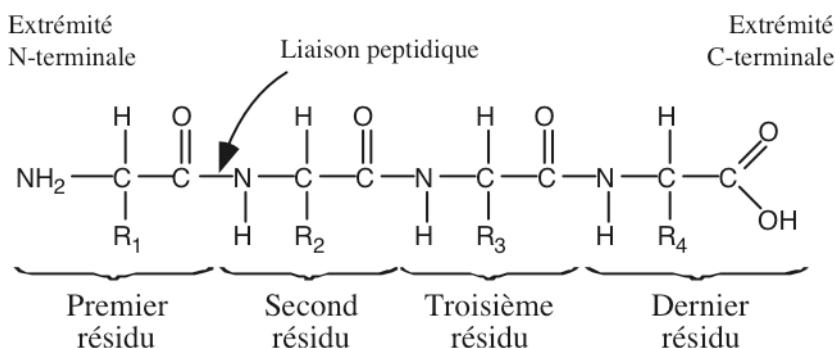


FIG. 1.2 – Un polypeptide formé de 4 acides aminés (tétrapeptide)

de Van der Waals - liaisons « hydrophobes » - sont des liaisons de type dipôle induit-dipôle induit : une asymétrie passagère de charge autour d'un atome (possible parce que la distribution électronique de charge autour d'un atome évolue avec le temps) induit une asymétrie opposée dans un atome adjacent ; ils s'attirent alors mutuellement. L'attraction est perdue lorsque les atomes s'approchent trop en raison de la répulsion provoquée par le recouvrement des nuages électroniques. Les liaisons de Van der Waals sont plus faibles que les liaisons hydrogène ou électrostatiques, mais elles sont efficaces en grand nombre. Avec les liaisons hydrogène de la chaîne polypeptidique, ces forces sont les principales responsables du repliement des protéines.

L'encombrement stérique joue également un rôle dans la structure des protéines car lorsque la chaîne protéique s'enroule, une chaîne latérale encombrante peut empêcher une courbure serrée. Les liaisons covalentes inter-résidus - sous forme de ponts disulfure entre deux résidus cystéine - permettent une meilleure stabilité pour les protéines qui sortent de l'environnement cellulaire (Brandon and Tooze, 1999).

2) les fonctions.

Les protéines ont des fonctions très diverses. Elles peuvent être des catalyseurs (enzymes), des transporteurs (hémoglobine,...), des hormones (effecteurs), des récepteurs (transduction d'un signal). Elles peuvent être des toxines ou des anticorps. Elles reproduisent le matériel génétique et participent à la structure de la cellule (cytosquelette, moteurs moléculaires). Toutes ces actions impliquent de manière générale la fixation d'une protéine sur une autre molécule (qui peut être aussi une protéine), *i.e.* l'interaction des protéines et des autres molécules. Cette fixation peut être plus ou moins spécifique. Comme indiqué précédemment, les chaînes latérales des acides aminés jouent le plus grand rôle dans ce phénomène général de fixation.

1.2 La structure des protéines

1.2.1 Détermination

1) La Radiocristallographie aux rayons X

La cristallographie aux rayons X est la méthode de choix pour résoudre la structure atomique des protéines. La première structure déterminée par cristallographie fut la myoglobine ; elle a été déterminée par Max Perutz et Sir John Cowdery Kendrew en 1958 (Kendrew et al., 1958) (Prix Nobel de Chimie en 1959). Cette méthode s'appuie sur la diffraction des rayons X par les électrons dans un cristal ; le principe est que les rayons X diffractés par les différents « plans » réticulaires du cristal seront en phase à différents angles d'incidence selon la longueur d'onde du rayonnement (obtention de la distance entre ces plans par interférence). Le nuage électronique des atomes lourds (la « densité électronique ») est déterminé par cette méthode. La seconde étape de la méthode consiste à placer les atomes correctement dans cette densité électronique par des techniques de modélisation moléculaire. La limitation la plus importante de la cristallographie est l'obtention d'un cristal de protéine, opération difficile voir impossible, notamment pour les protéines membranaires ou les protéines flexibles.

2) La Résonance Magnétique Nucléaire

La Résonance Magnétique Nucléaire (RMN) permet d'obtenir la structure des protéines en solution. Plusieurs paramètres géométriques peuvent être calculés sur la structure protéique, comme des angles de torsion par la mesure du couplage scalaire, ou bien des distances entre atomes (hydrogènes généralement) via la mesure du couplage dipolaire (Nuclear Overhauser Effect ou NOE). Ces dernières mesures étant sensibles à la dynamique de la protéine, elles peuvent donner une indication sur celle-ci. Les contraintes de distances entre atomes obtenues par RMN sont utilisées lors de calculs de dynamique moléculaire afin d'approcher la structure de la protéine (et sa dynamique).

3) La Protein Data Bank

La banque de données recensant les structures de protéines (et d'acides nucléiques et de quelques sucres) est la PDB (*Protein Data Bank* ou « *Brookhaven* » *Data Bank*) (Bernstein et al., 1977; Berman et al., 2000). Cette banque est extrêmement redondante, beaucoup de structures proviennent de protéines très proches (par exemple plus de 600 structures du lysozyme sont présentes dans la PDB). Il y avait 33 065 structures dans la PDB le 12 octobre 2005, dont 31 500 structures de protéines. Il est possible que la majeure partie des protéines uniques soient de nouveaux replis (évidemment, ceci n'est vrai que pour les protéines uniques à une date donnée et pas pour le total cumulé). L'augmentation du nombre de protéines uniques est plus ou moins linéaire (voir figure 1.3). Ceci pourrait changer avec les programmes de cristallographie à grande échelle qui cherchent maintenant à déterminer les structures des protéines dont aucun homologue n'est identifié dans la PDB.

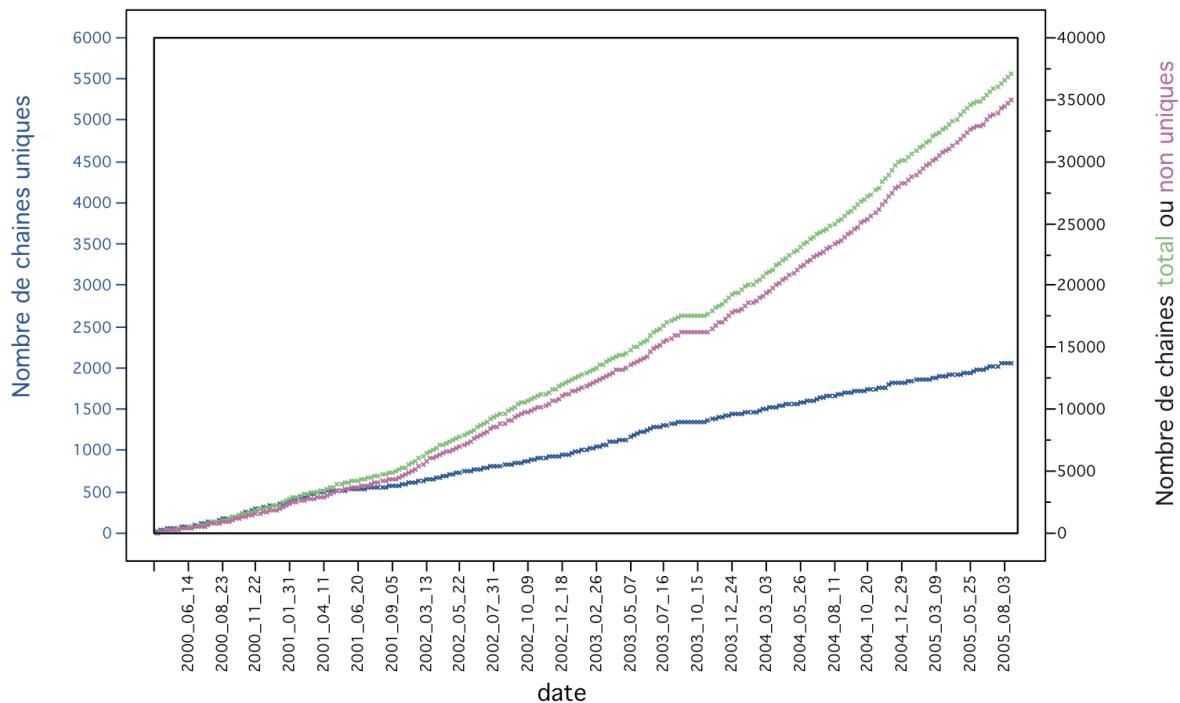


FIG. 1.3 – Statistiques sur les structures de la PDB. En vert, le total cumulé des structures de la PDB, en rouge, les structures non uniques, c-à-d. ayant au moins un homologue (déterminé à l'aide des séquences) dans la PDB et en bleu le total cumulé des protéines uniques, c-à-d. n'ayant pas d'homologue dans la PDB (l'échelle est différente pour ces structures). Ces données numériques sont extraites du site de la PDB (<http://www.rcsb.org>).

4) Le repliement

Une protéine pourrait adopter plusieurs conformations (voire un grand nombre), mais la plupart se replient spontanément dans une forme stable particulière et unique. Cette forme particulière est due au fait que les groupes du squelette peptidique et les chaînes latérales interagissent entre eux et avec l'eau. Ainsi, certaines conformations ont plus d'interactions stabilisantes que d'autres et sont donc favorisées (Alberts et al., 1994). Le paradigme du rapport entre la séquence protéique et sa structure (3D) tridimensionnelle provient des études de Christian Anfinsen sur la ribonucléase (Anfinsen, 1973). C. Anfinsen a déterminé qu'une protéine repliée aléatoirement est inactive, et que les protéines isolées en solution peuvent retrouver leur conformation active originale après dénaturation. La conclusion était donc que toute l'information nécessaire au repliement d'une protéine devait être inhérente à l'ordre de ses acides aminés (Alberts et al., 1994). D'autres études ont également tiré les mêmes conclusions, menant à la théorie générale que la séquence des acides aminés d'une protéine spécifie sa conformation (Stryer, 1994). Il est intéressant de noter que sur l'ensemble des séquences possibles, la population des différents repliements est hétérogène, 9 repliements représentent 46% des protéines d'une base de données non redondante (Thornton et al., 1995).

Les protéines appelées globulaires se replient en une forme globulaire compacte (contrairement aux protéines fibreuses ou aux protéines membranaires). La nature hydrophobe de certains acides

aminés rend ce repliement compact nécessaire (voir figure 4)). En effet, les chaînes latérales des acides aminés peuvent être polaires ou non polaires. Les résidus non polaires sont hydrophobes et groupés ensemble à l'intérieur de la structure globulaire de la protéine - pas de contact avec l'eau environnante -, tandis que les résidus polaires, et les groupes polaires du squelette, sont hydrophiles et forment des liaisons hydrogène avec l'eau ou les uns avec les autres. Ces liaisons hydrophobes et liaisons hydrogène sont responsables en grande partie de la stabilité de la structure protéique. Quand des liaisons hydrogène se forment entre des groupes du squelette peptidique, le groupe α -amide est le donneur et le groupe α -carboxylique l'accepteur.

La durée réelle du mécanisme de repliement observée (de l'ordre de la seconde) est de plusieurs ordres de grandeurs en dessous du temps nécessaire à l'exploration toutes les structures possibles (Robson, 1999). Certaines hypothèses sont donc proposées. Au début du repliement les éléments de structures secondaires se formeraient rapidement et s'organiseraient globalement de manière correcte. La protéine adopterait alors une conformation ouverte et flexible, appelée « globule en fusion » (*molten globule*), qui serait le point de départ pour un processus relativement lent pendant lequel les chaînes latérales seraient à plusieurs reprises ajustées sur la forme correcte de la structure tertiaire (les structures secondaires et tertiaires sont expliquées ci-dessous). Plusieurs chemins seraient possibles pour arriver à la conformation finale (Richards, 1991). Ce processus peut être résumé ainsi : repliement local, formation des interactions de longue portée, les réarrangements locaux menant ensuite à la conformation finale.

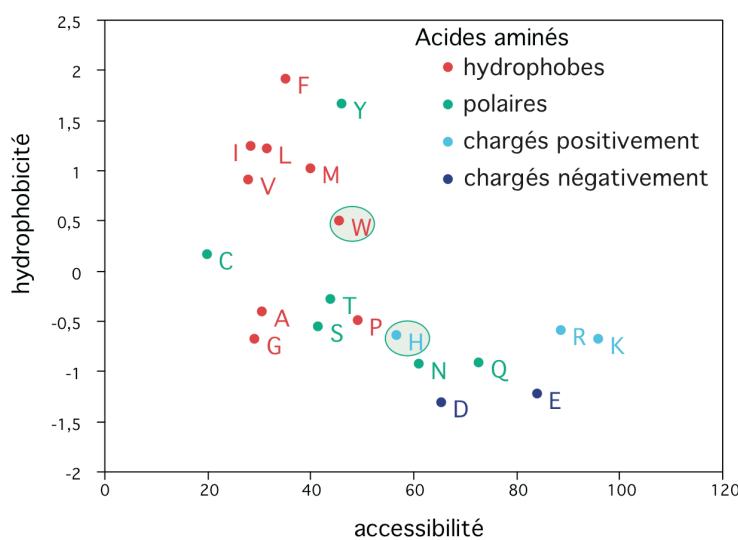


FIG. 1.4 – Hydrophobicité (selon l'échelle de Kyte et Doolittle (Kyte and Doolittle, 1982) en fonction de l'accessibilité au solvant des résidus, en \AA^2).

En résumé, en environnement aqueux, le repliement des protéines est conduit par la tendance des résidus hydrophobes à être exclus de l'eau. Les résidus et les groupes polaires du squelette peuvent interagir entre eux dans le cœur hydrophobe et avec l'eau à l'extérieur (voir figure 4)). Le repliement des protéines destinées aux environnements non aqueux, par exemple les protéines membranaires, diffère car les résidus non polaires ne doivent plus forcément se trouver dans un cœur hydrophobe.

Les forces et les interactions décrites ci-dessus permettent le repliement d'une protéine dans une conformation 3D fonctionnelle, et, ainsi que Chothia le précise, « la structure apparemment complexe des protéines est régie par un ensemble de principes relativement simples » (Chothia, 1984). La conformation s'établit grâce à plusieurs niveaux qui se construisent l'un sur l'autre (Koch et al., 1992). Ces niveaux se nomment structures primaire, secondaire, super-secondaire, tertiaire, et quaternaire, et sont décrits ci-dessous. D'autres niveaux de description sont aussi employés comme les domaines. Les domaines sont les unités compactes, souvent fonctionnelles et globulaires, d'une protéine, séparées par des sections plus flexibles de la chaîne protéique (Wetlaufer, 1973).

5) Organisation de la structure des protéines.

Quatre niveaux d'organisation pour les protéines sont traditionnellement définis :

structure primaire : séquence d'acides aminés ;

structure secondaire : répartition de certaines structures locales régulières nommées structures secondaires (hélice α et feuillet β) ;

structure tertiaire : organisation dans l'espace à trois dimensions des atomes de la protéine (ou organisation spatiale et interaction des éléments de structure secondaire) ;

structure quaternaire organisation spatiale de monomères (pour les protéines multimériques).

D'autres niveaux sont souvent ajoutés à cette hiérarchie. Par exemple, on définit les structures super-secondaires (*supersecondary structures*) ou les domaines (voir ci-après).

Les structures secondaires sont des conformations locales et périodiques présentes dans les protéines. Les hélices *alpha* et les feuillets *beta* - ces derniers étant associés par paires ou plus de manière parallèle ou anti-parallèle - en sont les plus fréquentes. Ces structures locales sont stabilisées liaisons hydrogène. Elles sont aussi nommées structures répétitives car elles sont caractérisées par une répétition de valeurs d'angles (ϕ, ψ ou (α, τ)) proches. L'hélice α la plus fréquente est une hélice droite comportant 3,6 résidus par tour. Elles sont stabilisées par des liaisons hydrogène entre le groupe *CO* du résidu i et le groupe *NH* du résidu $i + 4$ (voir figure 1.5(a)). D'autres hélices droites existent telles les hélices 3-10 et les hélices *pi* (figure 1.5(b)). Les hélices gauches sont très peu fréquentes. Dans les feuillets β , les liaisons hydrogènes établies mettent en jeu des régions distantes. Les brins β sont donc généralement organisés en feuillets β parallèles ou antiparallèles (voir figure 1.6), mais des coudes « *beta* » peuvent aussi se former. Les résidus qui n'appartiennent pas à une structure locale de ces types sont dits organisés en boucle. Les termes d'hélices α et brins (ou feuillets) β seront par la suite souvent regroupés sous le terme *Structural Secondary Elements* (SSE).

Les structures super-secondaires (*supersecondary structures*) (Rao and Rossmann, 1973; Richardson, 1981) sont des organisations de structures secondaires récurrentes dans les protéines. Elles se situent donc entre les niveaux structure secondaire et tertiaire. Certaines des plus connues sont : les motifs $\beta\alpha\beta$ où deux brins β parallèles sont reliés par une hélice α , les motifs en épingle à cheveux (β *hairpin* et α -*helix hairpin*).

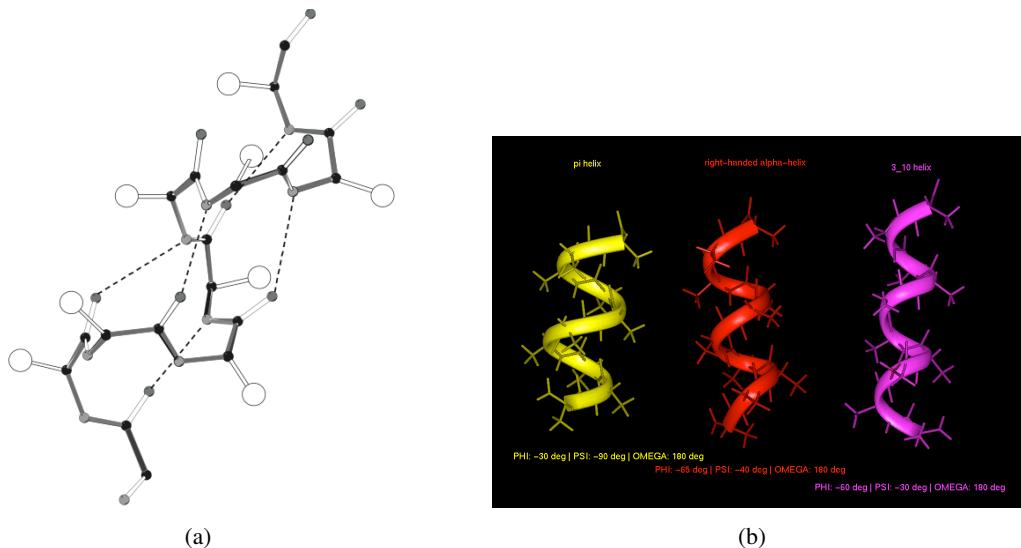


FIG. 1.5 – (a) Hélices α . Les traits pointillés indiquent les liaisons hydrogène. Figure extraite de la thèse de V. Escalier (Escalier, 1997). (b)Trois types d'hélices. Figure extraite du site IMB (<http://www.imb-jena.de/IMAGE.html>)

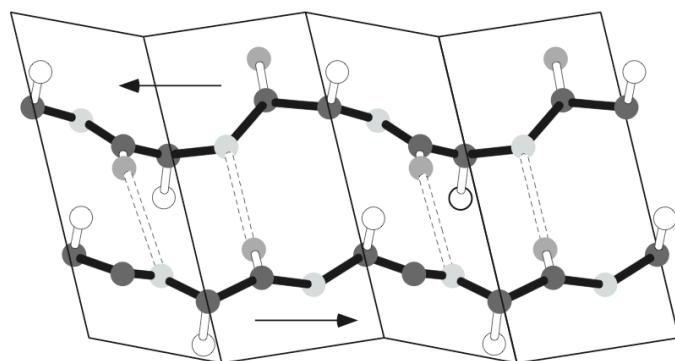


FIG. 1.6 – Brins et feuillets β . Les traits pointillés indiquent les liaisons hydrogène. Les chaînes latérales (R) se répartissent de part et d'autre du feuillet. Le feuillet représenté ici est antiparallèle. Figure extraite de la thèse de V. Escalier (Escalier, 1997).

Entre ce dernier niveau et les structures tertiaires se trouvent aussi les domaines. La définition du terme domaine est délicate n'est pas consensuelle. Un domaine structural est défini comme étant une unité compacte qui pourrait être stable indépendamment même si cela n'a pas été prouvé expérimentalement (Rose, 1979). Néanmoins, les résidus d'un domaine structural partagent plus de contacts entre eux qu'avec ceux d'autres domaines. Le terme de domaine fonctionnel est lié à la notion de site fonctionnel (par exemple, site de fixation d'une molécule). Les domaines fonctionnels sont plus restreints en terme d'étendue et de nombre de résidus et peuvent être localisés à l'interface des domaines structuraux.

1.2.2 Description géométrique du squelette

1) Les coordonnées cartésiennes

La description la plus simple du squelette est évidemment la liste des coordonnées cartésiennes de ses atomes ou de ses C_α . Cette description est celle sur laquelle s'appuie la mesure du RMSD entre deux structures superposées (voir section « Les mesures de similarité : les RMSD » page 32).

2) Les distances internes

La liste des distances internes entre atomes ou C_α du squelette peptidique permet de définir une mesure de conformation indépendante de l'origine des coordonnées. Les distances internes d'une sous-structure donnent donc une mesure « locale » de la conformation. Cette mesure permet évidemment les comparaisons sans avoir à superposer les coordonnées cartésiennes des structures concernées (voir section « Les mesures de similarités »).

3) Les angles *Phi*, *Psi* et *Omega*

Comme la liaison peptidique $-CO-NH-$ est plane, le squelette peptidique peut être décrit par les angles ϕ , ψ et ω , qui sont les angles dièdres de la chaîne tournant respectivement autour des liaisons $N-C_\alpha$, $C_\alpha-C$, et $C-N$ (voir figure 1.7 L'angle ω étant en général « trans » - (valant 180°), sauf dans les prolines où il est typiquement « cis »). Les études ne prennent en compte que le couple (ϕ, ψ) (MacArthur and Thornton, 1996). De plus, dans ce couple, ϕ et ψ ne peuvent pas prendre n'importe quelle combinaison de valeurs (Ramachandran et al., 1963). En outre, les angles ψ et ω sont très corrélés (Esposito et al., 2005).

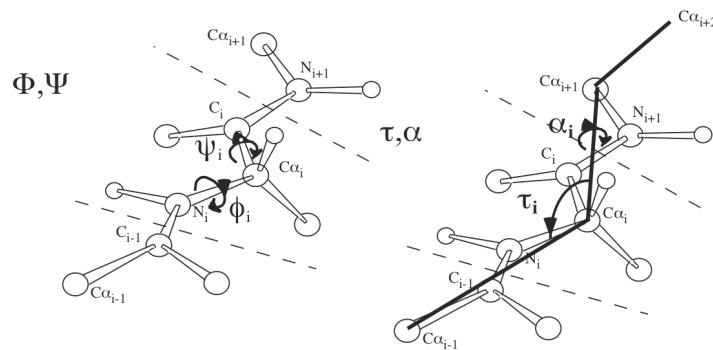
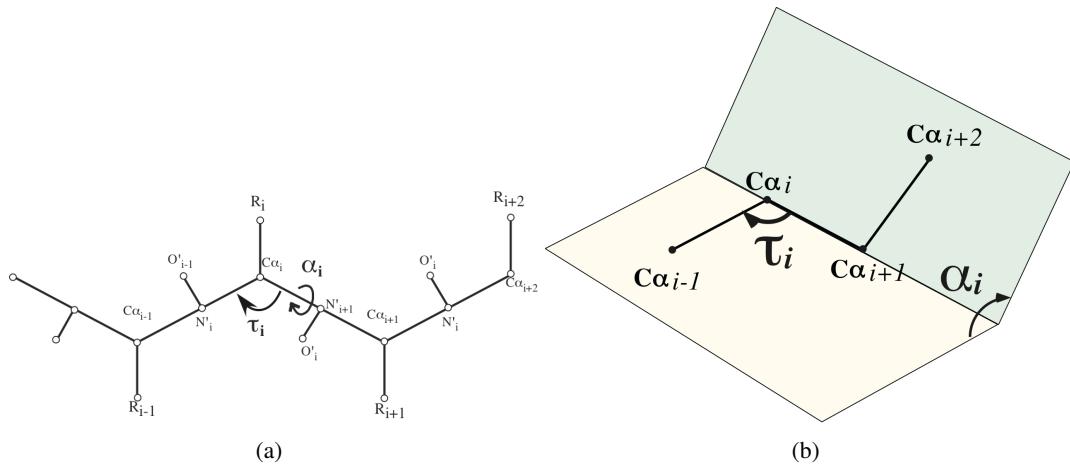


FIG. 1.7 – Les angles (ϕ, ψ) et (α, τ)

4) Les angles *Alpha* et *Tau*

L'angle α est l'angle dièdre défini par quatre C_α successifs et l'angle τ est l'angle plan défini par trois C_α successifs (figure 1.8). L'angle α_i est associé au deuxième des quatre C_α (le $C_{\alpha i}$) et l'angle τ_i est donc celui formé par les trois premiers C_α .

FIG. 1.8 – Les angles (α, τ)

Les angles α et τ ont été décrits par Levitt (Levitt and Warshel, 1975; Levitt, 1976). Toutefois, ils ont été utilisés précédemment par P.J. Flory lors de son étude de la conformation de polypeptides en pelotes statistiques (*random-coil*) (Flory, 1969).

Les liaisons $N - C_{\alpha_i}$ et $C_{\alpha_{i+1}} - C$ sont parallèles à $\pm 10^\circ$, et le groupe peptidique ($-C_{\alpha_i}HNH_2 - COOH$) est plan, il est donc possible de remplacer les angles ϕ et ψ par une seule valeur, l'angle α dont la relation avec ϕ et ψ est (Levitt, 1976) :

$$\alpha_i = 180^\circ + \phi_{i+1} + \psi_i + 20^\circ(\sin\phi_i + \sin\phi_{i+1}) \quad (1.1)$$

Comme les axes des liaisons $N - C_{\alpha_i}$ et $C_{\alpha_{i+1}} - C$ sont parallèles mais non colinéaires, l'angle τ , formé par 3 C_α successifs, varie avec l'angle α selon la relation :

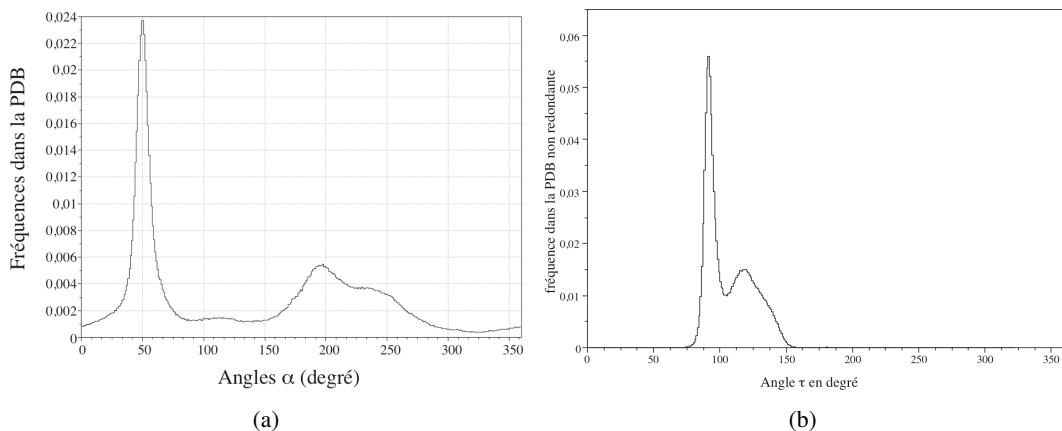
$$\tau_i = 106^\circ + 13^\circ \cos(\alpha_i - 45^\circ) \quad (1.2)$$

L'angle τ varie donc peu autour de 106° , comme le montre sa distribution dans la PDB (figure 1.9(b)).

Comme celle des angles (ϕ, ψ) , la distribution des angles (α) n'est pas uniforme (Oldfield and Hubbard, 1994). Les éléments de structures secondaires sont décrits par une suite répétitive d'angles α : une hélice α est une suite d'angles α proche de 50° tandis qu'un brin β est une série d'angles α fluctuant autour de 200° (entre 180° et 240° , figure 1.9(a)).

1.2.3 Autres propriétés géométriques

Il est possible de définir d'autres propriétés ou mesures géométriques des résidus ou des protéines et de les utiliser à des fins de comparaison. Les principales sont l'accessibilité au solvant définis par la surface de *i.e.* l'enveloppe décrite par une sphère roulant sur la surface des rayons de Van der Waals des atomes de la protéine.

FIG. 1.9 – Distribution des angles α et τ

1.3 La prédition de la structure des protéines

Trois grands types de méthodes sont utilisées pour la prédition de la structure 3D des protéines : la modélisation par « homologie », la reconnaissance de repliements., les méthodes de prédition *ab initio*.

1.3.1 La modélisation par « homologie »

La modélisation par « homologie » s'appuie sur l'hypothèse que des protéines homologues ont en général conservé la même fonction et une topologie et une structure proches. Pour établir l'homologie entre une séquence et une structure connue, il faut qu'elles présentent une similarité suffisante au niveau de leur séquence en acides aminés après alignement.

1.3.2 La reconnaissance de repliements

Les méthodes de reconnaissance de repliements ou *threading* reposent sur l'observation que les structures 3D sont mieux conservées que les séquences (Chothia and Lesk, 1986) et qu'ainsi les structures constituent un meilleur support pour caractériser des protéines homologues distantes que leur séquences. De plus, le nombre de repliements différents estimé est de plusieurs ordres de grandeur en dessous du nombre de familles de séquences différentes (dégénérescence structure-séquence). Ce nombre est estimé de 1000 à 8000 selon les études (Chothia, 1992; Orengo et al., 1994; Wang, 1996; Wang, 1998; Govindarajan et al., 1999).

Les méthodes de threading (Smith et al., 1997) visent donc à calculer l'adéquation entre une séquence de structure inconnue et une structure 3D connue (ou parties de celle-ci). La séquence est « enfilée » sur chaque repliement d'une librairie de cœurs représentatifs de la PDB. La compatibilité de la séquence avec un repliement particulier peut se mesurer grâce à un potentiel empirique : potentiels statistiques de distances entre résidus, précalculés sur la PDB (Sippl, 1990) ou scores de paires

de résidus étant donné la structure secondaire où se trouvent ceux-ci.

La superposition des structures d'une même famille structurale permet de caractériser les régions conservées utiles pour la reconnaissance de repliements.

1.3.3 La modélisation *ab initio / de novo*

Les méthodes *ab initio* sont fondées sur le paradigme que l'état natif d'une protéine est à son minimum global d'énergie libre et cherchent à mener une recherche de ce minimum par un parcours à grande échelle de l'espace conformationnel. Les clés de ces méthodes sont d'avoir une fonction « énergie » adéquate et une procédure efficace pour parcourir l'espace des conformations. Pour cette raison, ces méthodes utilisent souvent un sous-ensemble des atomes de la chaîne peptidique et des chaînes latérales. La fonction énergie employée doit donc refléter les atomes omis et le solvant. Une des méthodes les plus connues et efficace, Rosetta (Simons et al., 1999), fait varier la structure de petits segments (3 et 9) entre des conformations compatibles avec la séquence locale mais ayant des conformations différentes (ces fragments sont précalculés sur les protéines connues de la PDB). Un grand nombre de conformations globales sont générées par combinaison de ces segments locaux (ce qui échantillonne l'espace conformationnel) et la fonction énergie est évaluée sur ces structures. Les prédictions *de novo*, à l'aide d'alphabets structuraux (de Brevern et al., 2001) - dont nous parlons plus loin - sont à classer parmi ces méthodes qui ne s'appuient pas sur une homologie particulière avec une structure mais plutôt sur un apprentissage de ce qu' « est » une structure (petits blocs chevauchants, corrélation de l'enchaînement de ces blocs).

1.4 L'annotation structurale

Le séquençage systématique des génomes demande le développement de méthodes d'annotation nouvelles afin de mieux reconnaître les gènes ayant divergé. Les méthodes usuelles de l'annotation s'appuient sur les biais statistiques liés aux gènes (biais de codage, biais d'usage du code), sur la reconnaissance de signaux génétiques (start, stop, promoteurs, sites de fixation du ribosome, *etc*), et surtout sur la similarité des phases ouvertes de lecture avec des séquences déjà caractérisées. Les outils de détection de similarités de séquence généralement utilisés pour établir cette similarité sont BLAST (Altschul et al., 1990; Altschul et al., 1997) et FASTA (Pearson and Lipman, 1988; Pearson, 2000). La dégénérescence bien connue entre séquences et structures (Chothia and Gerstein, 1997) limite ces méthodes puisqu'il sera difficile de détecter la similarité entre séquences ayant divergé depuis longtemps. Ainsi, réglés à un seuil de détection de 1% de faux positifs, BLAST ou FASTA ne reconnaissent comme significatives que 15% de séquences apparentées et possédant moins de 40% de résidus conservés entre elles (Brenner et al., 1998). La détection de similarités est améliorée par des méthodes basées sur des alignements multiples (Park et al., 1998), ou « profils », comme PSIBLAST (Altschul et al., 1997).

Pour aller plus loin dans la détection de similarités, les méthodes de reconnaissance de replie-

ments. sont des outils de grand intérêt. Elles permettent de détecter des similarités indécelables grâce aux seules séquences.

CHAPITRE 2

Comparaison de deux structures

2.1 Introduction

La structure 3D des protéines est un domaine d'étude important car la conformation d'une protéine joue un rôle essentiel dans sa fonction biologique. Les techniques de comparaison de structures protéiques sont donc essentielles dans beaucoup de domaines de recherche, notamment dans la prédiction de la structure d'une protéine et dans la compréhension de l'évolution des structures protéiques.

2.1.1 Les premières comparaisons de structures

Les premières méthodes de comparaison de structures protéiques datent des années 1970. Comme la comparaison de la position des atomes par rapport à la densité électronique fait partie du processus de détermination de la structure par diffraction des rayons X, les premières méthodes étaient dédiées à la comparaison d'une structure avec elle-même. Dans le cas de structures différentes, voire très différentes, le problème du choix de la correspondance des éléments à comparer est le principal problème.

En effet, comme deux protéines ne sont pas composées de la même séquence d'acides aminés, il n'est pas possible de comparer tous les atomes, chaînes latérales comprises. Il faut donc se restreindre au moins aux atomes du squelette peptidique. Cependant, certains atomes des chaînes latérales jouent un rôle très important par exemple dans la fixation d'un substrat ou d'un co-effecteur et inclure ces atomes dans la comparaison est donc utile. En outre, l'organisation des structures secondaires semble conservée dans les protéines - même lointaines - d'une même famille. Ne comparer que celles-ci serait donc plus pertinent, au moins pour des protéines assez différentes. Nous avons alors trois niveaux de représentation : une représentation « tout atome », une représentation restreinte aux atomes du squelette peptidique et une représentation en termes de structures secondaires.

Comme en comparaison de séquences, l'hypothèse sous-jacente à la comparaison de structures est généralement l'homologie supposée de ces structures (ou d'une de leurs sous-structures). Ainsi, on cherche à mettre en correspondance la position des acides aminés, et on tient compte habituellement de leurs séquences et de sa divergence. Dans les cas où l'on veut mesurer la convergence éventuelle de structures ou de sites, la comparaison ne doit pas prendre en compte la séquence des résidus.

A chaque niveau de représentation correspondent plusieurs représentations possibles des éléments. Il existe des centaines d'articles portant sur la comparaison de structures, différent non seulement sur la représentation utilisée mais aussi sur la méthode de comparaison. Assez peu cependant résument l'ensemble des méthodes. W. Taylor et col. ont publié quelques articles très complets sur les méthodes d'alignement de deux structures (Brown et al., 1996; Eidhammer et al., 2000; Taylor et al., 2001), dont certains éléments sont repris et mis à jour dans (Wang, 2005). D'autres articles résument certaines des méthodes (Holm and Sander, 1994c) mais les plus nombreux sont les articles testant et comparant les résultats de plusieurs méthodes (rarement plus de 6 méthodes à la fois) (May, 1999; Sierk and Pearson, 2004; Novotny et al., 2004; Kolodny et al., 2005).

2.1.2 Définitions

En général, un élément d'une structure est mis en correspondance avec un seul élément de l'autre structure. Plus formellement, A et B étant deux structures respectivement de longueur n et m , décrites par les éléments $A = (a_1, \dots, a_n)$ et $B = (b_1, \dots, b_m)$:

1. une **correspondance ou équivalence** est un ensemble de paires tel que $P(A, B) = \{(a_{i_1} b_{j_1}), (a_{i_2} b_{j_2}), \dots, (a_{i_k} b_{j_k})\}$. Les résidus sont donc mis en correspondance ou équivalence ;
2. un **alignement M** pour A et B est une correspondance qui satisfait deux conditions : $i_1 < i_2 < \dots < i_k$ et $j_1 < j_2 < \dots < j_k$. On $M(A)$ notera le sous-segment $(a_{i_1}, \dots, a_{i_k})$, de même pour $M(B)$;

2.1.3 Objectifs de ces méthodes

Les objectifs des méthodes de comparaison structurale sont en général de calculer une mesure quantitative de similarité entre deux structures protéiques et/ou de générer un alignement structural, souvent converti en alignement des séquences (les correspondances qui ne respectent pas la séquentialité sont plus rares).

Les quatre points importants d'une méthode de comparaison structurale (tels qu'ils ont été définis dans l'article de L. Holms (Holm and Sander, 1996c)) sont :

- **la représentation des structures** : les structures protéiques sont toujours simplifiées mais les caractéristiques conservées doivent être suffisantes pour la comparaison. Exemples : les C_α sont décrits par leur coordonnées cartésiennes, ou par leurs distances internes, les SSE sont décrits par des vecteurs, *etc* ;
- **la mesure de similarité ou de dissimilarité** : il faut pouvoir déterminer si un sous-alignement est meilleur qu'un autre pendant le processus d'alignement. Cette mesure est bien sûr totalement dépendante de la représentation ;
- **l'algorithme de comparaison** : il peut être soit un algorithme général déjà connu (recherche des cliques maximales dans un graphe, Monte Carlo,...) ou un algorithme *ad hoc* ;
- **les post-traitements** : par exemple le calcul d'un score exprimant la significativité des résultats (il peut être empirique ou statistique, humain ou automatique).

La représentation des structures

Les niveaux de représentation sont, du plus fin au plus global :

- les atomes, généralement seuls certains atomes sont sélectionnés par exemple ceux du site actif ;
- les résidus *i.e.* les atomes du squelette peptidique, souvent représentés uniquement par les C_α ;
- les structures secondaires ou d'autres fragments structuraux ;
- la forme générale de la molécule.

Le niveau atomique et celui du squelette peptidique sont distincts car, même si dans les deux cas la comparaison joue sur des atomes, dans le premier cas ce sont les atomes qui sont mis en correspondance et dans l'autre cas ce sont les résidus.

La structure des protéines est toujours comparée d'abord d'un point de vue géométrique par exemple par comparaison un ensemble de points ou de vecteurs. La représentation géométrique ou spatiale de ces points ou vecteurs est assez constante mais d'autres informations peuvent être ajoutées : l'ordre de ces éléments dans la séquence protéique, les propriétés physico-chimiques des acides aminés, leur type, leur charge, leur accessibilité au solvant, les liaisons hydrogène ou autres... Les liaisons hydrogènes, ou tout autres propriétés impliquant deux acides aminés, sont dites relationnelles. Au lieu de classer les informations en géométriques et non géométriques, il est possible de les classer en propriétés individuelles (coordonnées cartésiennes, charge...), et propriétés relationnelles (distances internes, liaisons chimiques...).

Les mesures de similarité/dissimilarité

Les mesures de similarité sont totalement dépendantes de la représentation. Elles seront décrites plus loin avec les méthodes.

L'algorithme de comparaison

Les méthodes de comparaison de structures sont essentiellement géométriques. Le problème de comparer deux structures revient donc à comparer plusieurs éléments géométriques (points, vecteurs, courbes...) ordonnés ou non. Si la correspondance n'est pas donnée, c'est un problème le plus souvent NP-complet (Lathrop, 1994). Il faut donc le circonscrire avec la méthode et/ou avec la mesure de similarité. Beaucoup de méthodes de comparaison d'ensembles de points non dédiées à la comparaison de structures protéiques existent mais les utiliser telles quelles n'est pas adapté car on ne tire alors pas parti des spécificités des protéines. Les algorithmes qui ont été adaptés pour la comparaison structurale sont : la programmation dynamique, certains algorithmes de la théorie des graphes (recherche de cliques maximales, recherche de graphes isomorphes), algorithmes d'optimisation/stochastiques (Monte Carlo, recuit simulé, algorithmes génétiques) ainsi que des algorithmes de classification. D'autres algorithmes sont dédiés à la comparaison de structures protéiques. Ces algorithmes ont donc été adaptés mais en général d'autres étapes - préliminaires ou postérieures - sont ajoutées, fournissant une procédure unique pour chaque méthode de comparaison.

Les « post-traitements »

Un score final est généralement calculé pour mesurer la similarité des deux structures comparées. Certaines méthodes sont dédiées à cet objectif et se passent même de la procédure d'alignement. Les méthodes de recherche de similarités structurales sur banque, c'est-à-dire. permettant la comparaison d'une structure protéique dite requête à toutes les structures présentes dans un banque de structures, nécessitent un classement final des structures cibles de la banque selon leur similarité avec la

requête. Dans ce but, un score de similarité est calculé et généralement un Z-score est aussi utilisé (voir plus loin).

Le plan de ce chapitre

Du fait du nombre élevé de méthode de comparaison structurales, il est indispensable de les regrouper par catégories. J'ai choisi de classer les méthodes de comparaison structurales selon le niveau de représentation, puis selon la représentation, puis selon le type de comparaison. Il n'existe en réalité qu'assez peu de types de comparaison possibles lorsque que le niveau et le mode de représentation ont été choisis. Un autre plan de séparation sera le classement des méthodes de comparaison en méthodes « locales » et « globales ». En effet il est possible soit de comparer tous les éléments en même temps (au besoin en supprimer certains...) soit de chercher les sous-groupes d'éléments similaires sans s'attacher à leur organisation générale. Evidemment, les méthodes sont rarement purement globales ou locales.

2.2 Description au niveau atomique

Les méthodes décrites dans cette section permettent rarement de comparer des protéines entières. Toutes les méthodes travaillant au niveau atomique ne seront pas exhaustivement examinées car la majorité d'entre elles sont des méthodes utilisées pour l'amarrage de molécules (*docking*).

Ces méthodes se classent en deux catégories : les méthodes comparant des atomes ou des groupes d'atomes et les méthodes comparant des surfaces. L'accent sera mis sur la première catégorie, car la seconde n'est jamais utilisée pour comparer des protéines entières mais pour comparer des sites spécifiques. Dans cette première catégorie, trois sous-catégories peuvent être distinguées : les méthodes fondées sur la théorie des graphes, les méthodes de recherche de motifs et les méthodes fondées sur le hachage géométrique (*geometric hashing paradigm*).

2.2.1 Graphes et comparaison structurale

Les graphes permettent une représentation des structures quel que soit le niveau de représentation (atomique, des résidus ou des structures secondaires). Je vais donc exposer ici brièvement les méthodes de comparaison fondées sur la théorie des graphes tout en restant dans le cas général. Les cas particuliers pour chaque niveau de représentation seront abordés ensuite lors de la description de ces niveaux.

Une structure peut toujours être représentée par un graphe, les sommets étant les « unités » à comparer (atomes, résidus représentés par les C_α , SSE...) et les arêtes représentant les relations entre les unités (distances, angles, types de liaison...). Certaines propriétés peuvent être associées aux sommets (type de l'atome, *etc.*).

Dans cette représentation, deux principaux types de méthodes existent pour comparer les structures : la recherche des plus grands sous-graphes isomorphes dans les deux graphes ou la recherche des cliques maximales dans le graphe de « correspondance » construit à partir des deux graphes. Les deux problèmes, recherche de sous-graphes isomorphes et recherche de cliques maximales sont NP-complets (Garey and Johnson, 1979).

Algorithme de Ullman

Une méthode pour la recherche de sous-graphes isomorphes a été décrite par Ullman (Ullmann, 1976) et a été utilisée par Sheridan (Sheridan et al., 1989; Pepperrell and Willett, 1991).

Soit deux graphes G_A et G_B . Soit recherchés les sous-graphes de G_B isomorphes à G_A , où chaque sommet du sommet de G_B est mis en équivalence avec un sommet de G_A . Les sous-graphes de G_B de taille N isomorphes à des sous-graphes de G_A permettent de construire ceux de taille $N + 1$. Pour ajouter un sommet, il faut que ses relations avec les sommets du sous-graphe de G_B soient similaires aux relations des sommets équivalents de G_A . Ces sous-graphes sont stockés dans un arbre de recherche qui est exploré en profondeur d'abord. La complexité de l'algorithme est factorielle. Une variante, présentée par A. Lesk (Lesk, 1979) diffère de la précédente en ce que les relations similaires sont re-

cherchées d'abord au lieu d'être vérifiées à chaque étape. Une autre variante - basée sur l'intersection de listes de sommets - a été développée par Jakes et Willet (Jakes and Willett, 1986). L'algorithme d'Ullmann a aussi été adapté à la comparaison des chaînes latérales par Poirrette (Poirrette et al., 1997).

Le second type de méthodes construit un graphe de correspondance à partir des deux graphes G_A et G_B (voir figure 2.1). Ce type de méthodes a été proposé par (Levi, 1972; Barrow and Burstal, 1976). Elles ont été appliquées à la recherche de sous-structures communes dans (Cone et al., 1977; Brint and Willett, 1987; Pepperrell and Willett, 1991; Takayashi et al., 1987; Subbarao and Haneef, 1991; Ho and Marshall, 1993)

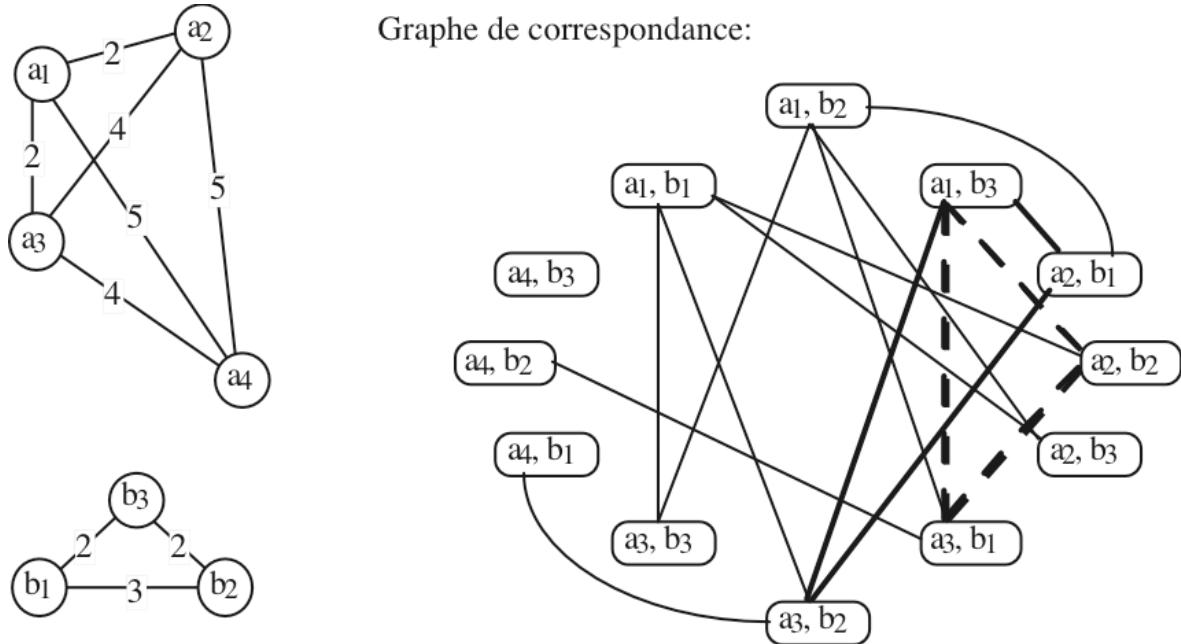


FIG. 2.1 – Le graphe de correspondance construit à partir de deux ensembles de points représentant chacun une structure (les deux graphes à droite). Pour qu'un paire (a_x, b_y) soit liée à une paire (a_i, b_j) , il faut que la différence des distances internes $d(a_x, a_i)$ et $d(b_y, b_j)$ soit inférieure ou égale à 1. Ainsi, un noeud contenant a_1 ne sera jamais lié à un noeud contenant a_4 car quelle que soit les éléments b la différence entre les distances est toujours au moins de 2. Le graphe de correspondance contient deux cliques de taille 3, indiquées en traits gras. Cette figure est extraite de la thèse de V. Escalier (Escalier, 1997).

SuMo

SuMo (Jambon et al., 2003) est un programme assez spécifique pour la recherche de sites actifs. Tous les atomes ne sont pas représentés dans la méthode SuMo, certains sont en effet regroupés pour former un à trois « groupes chimiques » par résidu. Plusieurs paramètres sont associés à ces groupes : la position moyenne des atomes, la position du centre de masse et la densité moyenne en atomes. Les triplets de groupes chimiques distants de moins de 8 Å forment des triangles. Plusieurs paramètres y sont associés comme la distance entre groupes, l'orientation du triangle (produit mixte des trois vecteurs ayant pour origine le centre de masse des atomes...). Un graphe est ensuite construit avec ces

triangles pour sommets et une arête quand 2 triangles sont adjacents, *i.e.* partagent deux sommets.

Un tel graphe est construit pour chaque structure. Pour comparer deux structures, les triangles similaires sont d'abord recherchés. Les triangles similaires ont les mêmes groupes chimiques, ainsi que des distances et une orientation similaires. Ces paires de triangles constituent les sommets du nouveau graphe de « comparaison ». Pour qu'une arête soit définie entre deux sommets (donc ici constitués de paires de triangles), il faut que les triangles soient connectés dans chacun des deux graphes représentant les structures et que les angles qu'ils forment dans chaque structure soient proches.

Cette méthode est rapide : 10 secondes de pré-traitement par protéine et 2 secondes pour la comparaison de protéines d'environ 250 résidus. Cependant, le serveur SuMo ne met à disposition non pas une comparaison de structures entières mais un service de recherche de site fonctionnels (sites actifs ou site de liaison d'un ligand). Il permet donc de chercher si une structure protéique donnée contient un des sites actifs de la banque de sites précalculés du serveur ou de chercher un site actif donné dans les structures protéiques de la PDB. Cette méthode n'est donc pas dédiée à la comparaison de structures entières.

2.2.2 Construction de motifs structuraux

Certaines méthodes ne s'appuient pas sur la théorie des graphes mais construisent des motifs de taille croissantes. Il s'agit notamment de celles de Crandell et coll. (Crandell and Smith, 1983) et de V. Escalier et coll. (Escalier et al., 1998). L'esprit de ces deux méthodes est similaire. Il est possible de comparer des protéines entières par Escan (même de manière multiple), néanmoins ces méthodes sont particulièrement adaptées à la recherche de petits « motifs » connus ou à la comparaison de petites molécules (de l'ordre de 30 atomes (Crandell and Smith, 1983)).

L'idée générale est de construire des « motifs » communs de tailles croissantes et la représentation des structures est donnée par les distances internes entre atomes. Les deux méthodes recherchent donc des paires de sous-ensembles d'atomes dont toutes les distances internes sont similaires - *i.e.* leurs différences sont inférieures à un seuil donné - dans les deux structures. La méthode de construction des ensembles d'atomes similaires dans les deux méthodes est toutefois un peu différente. La méthode de Crandell ajoute un atome à chaque ensemble déjà formé de taille k puis cherche si les nouveaux ensembles de taille $k+1$ sont présents dans les deux structures comparées. Dans la méthode Escan, les ensembles de taille $k+1$ sont construits à partir de deux ensembles de taille k ayant $k-1$ éléments en commun, ce qui ne nécessite que la vérification d'une seule distance interne - celle entre les k^{e} atomes (voir figure 2.2). Il est à noter que les atomes peuvent avoir une « couleur » qui entrera dans la spécification des similarités (par exemple le type de l'atome).

Une autre différence de la méthode Escan avec celle de Crandell, est qu'après que des motifs de taille suffisante ont été trouvés (une limite de distance entre atomes peut être fixée), ces fragments structuraux locaux sont assemblés pour former de plus gros fragments par la technique du *branch and bound*. Lors de cette seconde étape, chaque ensemble est le sommet d'un graphe et il y a une arête entre deux sommets si les deux ensembles sont compatibles, c'est-à-dire si les équivalences des

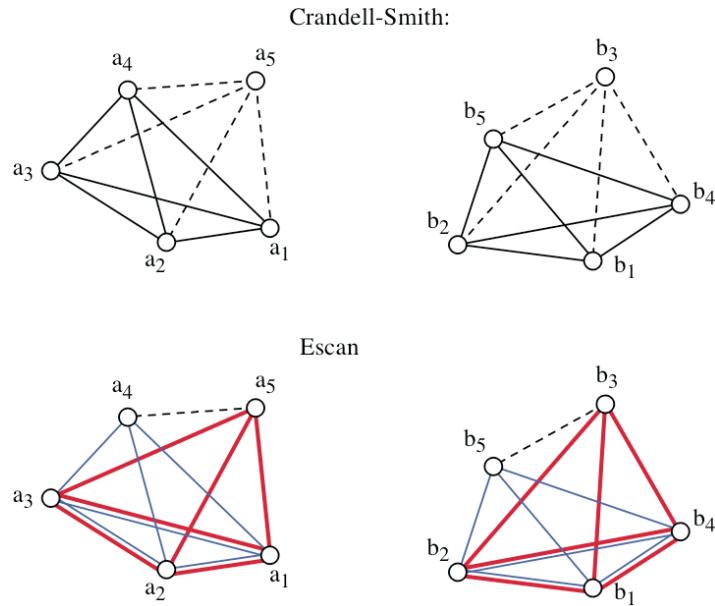


FIG. 2.2 – **Méthode de Crandell et Smith** : construction de deux sous-ensembles de taille k (ici 5) à partir d'ensembles similaires de taille $k - 1$. Ces deux sous ensembles sont conservés si toutes les 4 distances en pointillé sont toutes similaires.

Méthode Escan : construction d'un sous-ensemble similaire de taille k à partir d'ensembles similaires de taille $k - 1$ ayant $k - 2$ éléments en communs. Ici, un ensemble de taille 5 est construit à partir d'ensembles de taille 4 ayant 3 éléments communs : soient deux paires d'ensembles similaires de taille 4 ($(a_1, a_2, a_3, a_4), (b_4, b_1, b_2, b_5)$) (rouge) et ($(a_1, a_2, a_3, a_5), (b_4, b_1, b_2, b_3)$) (bleu) avec a_i le i^{e} atome de la protéine A et b_i le i^{e} atome de la protéine B. Ces deux ensembles ont en commun ($(a_1, a_2, a_3), (b_4, b_1, b_2)$). Ils formeront l'ensemble de taille 5 ($(a_1, a_2, a_3, a_4, a_5), (b_4, b_1, b_2, b_5, b_3)$) si la distance entre a_4 et a_5 est similaire à la distance entre b_5 et b_3 . Ce processus est répété pour toutes les tailles jusqu'à atteindre la taille maximale et à chaque, seule une distance doit être vérifiée.

atomes communs sont les mêmes et si les différences de distances internes restent en dessous du seuil. Un algorithme de *branch and bound* est utilisé pour trouver les cliques maximales de ce graphe.

Un article récent (Parthasarathy and Coatney, 2002; Coatney and Parthasarathy, 2003) décrit aussi une méthode de construction de motifs par taille croissante. Elle s'appuie sur une première étape de disqualification des paires d'atomes en dehors d'une distance limite (étape déjà présente dans la méthode Escan), heuristique qui diminue grandement la complexité réelle de l'algorithme. La méthode exposée semble la même que celle utilisée dans Escan, mais les auteurs ne citent ni la méthode Escan ni celle de Crandell...

programmes SPASM et RIGOR

Dans ces deux programmes (Kleywegt, 1999), tous les atomes ne sont pas représentés : un pseudo-atome est défini pour les chaînes latérales et seul le C_α est pris en compte pour la chaîne principale. SPASM recherche les occurrences d'un motif donné dans les structures protéiques et RIGOR recherche les motifs présents dans une structure donnée. Les deux fonctionnalités sont donc les mêmes que celles mises à disposition sur le serveur de SuMo.

La méthode utilisée dans SPASM est fondée sur un algorithme de recherche « en profondeur d'abord » décrit par Kleywegt et coll. (Kleywegt et al., 1989; Kleywegt and Jones, 1997). Le motif

doit être défini par l'utilisateur, il peut contenir les C_α et/ou les pseudo-atomes des chaînes latérales et de l'information de séquence (avec des substitutions autorisées). D'autres contraintes sont ajoutées comme la séquentialité, une longueur de *gap* identique dans les deux structures, des résidus voisins identiques. Dans une première étape, tous les résidus d'une structure du même type qu'un des résidus du motif sont cherchés. Ensuite, les combinaisons de résidus ayant une organisation similaire à ceux du motif sont recherchés et construits récursivement. Si une combinaison de résidus - *i.e.* un motif - atteint le nombre de résidus présents dans le motif, le $RMSD_c$ (voir section Les mesures de similarité : les RMSD) est calculé pour tester si les deux motifs sont vraiment similaires.

Le second programme RIGOR recherche un motif donné dans une structure. La mise en place d'une banque de motifs structuraux a donc été nécessaire. La banque établie n'est pas une banque complète mais a été construite à partir des informations contenues dans les fichiers de la PDB et dans la littérature. Seuls certains motifs ont été sélectionnés par rapport à quelques critères de séquence (résidus intéressants, répétitions, hydrophobes...).

2.2.3 Hachage géométrique

Le hachage géométrique utilisé dans le domaine de l'analyse d'images a été introduit par Lamdan et coll. (Lamdan et al., 1988; Lamdan and Wolfson, 1988). Il a été utilisé pour la comparaison de molécules par Fischer, Nussinov et Wolfson (Nussinov and Wolfson, 1991; Fischer et al., 1992; Bachar et al., 1993; Fischer et al., 1994). La méthode repose sur l'idée que si deux sous-ensembles d'atomes de deux structures A et B sont semblables, géométriquement, il est possible de trouver un repère où exprimer les coordonnées des points de A, et un repère où exprimer les coordonnées de B, et dans lesquels les coordonnées de chaque point des sous-ensembles semblables seront proches.

Etant donné certaines règles de construction géométrique, il est possible de déterminer un repère de manière unique à partir d'un ensemble ordonné de trois points non alignés. Pour chaque triplet de points de A (ou de B), un repère dit local est défini, les coordonnées des autres points de A (ou B) sont exprimées dans ce repère. Ces données (repères, coordonnées) sont stockées dans une table de hachage. Un score d'appariement par triplet, qui est le nombre de points de B ayant les mêmes coordonnées locales qu'un point de A, peut ensuite être rapidement calculé avec l'utilisation de cette table de hachage. La transformation rigide qui superposera deux triplets des 2 structures superposera aussi les points de mêmes coordonnées locales dans les deux triplets. Cette même opération est effectuée pour tous les couples de repères locaux dont le score d'appariement dépasse un certain seuil, ce qui permet de construire un ensemble de triplets composés d'un point de A, d'un point de B, et d'une transformation rigide. Un algorithme de classification permet ensuite de regrouper les triplets ayant des transformations rigides proches. Les classes obtenues à l'issue de cette étape sont les sous-structures communes cherchées.

La complexité de l'algorithme se calcule de la manière suivante : pour le remplissage de la table de hachage, on calcule les coordonnées de chaque point de A dans chaque repère local. Chaque repère étant construit à partir d'un triplet de points, cette étape nécessite $O(m^4)$ opérations, où m est le

nombre de points dans A. De même, l'opération suivante, qui consiste à rechercher dans la table les coordonnées locales des points de B pour chaque repère local, nécessitera $O(n^4)$ opérations si l'on suppose que le nombre de collisions dans la table de hachage est borné par une constante (n est le nombre de points dans B). Les auteurs utilisent ensuite une méthode de classification dont la complexité est quadratique. La complexité de l'algorithme est alors déterminée par la complexité de la première étape, soit $O(m^4 + n^4)$.

2.2.4 Méthodes fondées sur la forme

Ces méthodes ne permettent pas de trouver les blocs structuraux communs à deux structures mais permettent cependant de comparer des structures. Ces méthodes sont basées sur l'analyse de Procruste (Rohlf and Slice, 1990) et utilisées en morphométrie où l'on cherche à déceler des homothéties. Elles utilisent des points d'ancrage précédemment définis sur les structures et comparent les structures en opérant non seulement une translation-rotation mais aussi une transformation affine des points d'ancrage (Wu et al., 1998b).

2.3 Description au niveau du squelette peptidique

Le squelette peptidique est très souvent décrit par les coordonnées cartésiennes et ces coordonnées sont souvent restreintes à celles des C_α . Pour comparer deux structures à l'aide de cette description, il faut effectuer une transformation rigide d'une structure sur l'autre. C'est pourquoi cette description peut être dite « externe » par opposition aux descriptions en coordonnées internes où les deux structures peuvent être comparées directement. Les descriptions en coordonnées internes peuvent être soit les distances internes, soit les angles dièdres (ϕ, ψ) ou (α, τ) soit d'autres repères définis par des éléments des structures. Les méthodes ajoutant d'autres informations à la structure seront abordées à la fin de cette section.

2.3.1 Les mesures de similarité : les RMSD

La première mesure de similarité structurale définie et la plus usitée est le RMSD (*Root Mean Square Deviation*) sur les coordonnées. Il s'agit de la racine carrée de la moyenne des distances entre les atomes mis en correspondance dans les deux structures - décrites par leurs coordonnées cartésiennes.

Ce $RMSD_c$ est donc :

$$RMSD_c = \sqrt{\frac{\sum_{i=1}^n (D(a_i, b'_i))^2}{N}} \quad (2.1)$$

où N est le nombre d'atomes mis en correspondance : dans la structure B l'atome b'_i est mis en correspondance avec l'élément a_i de la structure A, et $D(a_i, b'_i)$ est la distance entre les atomes a_i et b'_i après superposition optimale de tous les atomes mis en correspondance (ensembles $M(A)$ et $M(B)$). Une superposition optimale de $M(A)$ et $M(B)$ est donc une transformation rigide T (une translation-rotation) telle que le $RMSD_c$ est minimal. Beaucoup de méthodes pour trouver cette transformation optimale ont été décrites. Certaines font appel à un formalisme fondé sur les quaternions (Kearsley, 1989; Zuker and Somorjai, 1989), certaines utilisent une diagonalisation de matrices (Kabsch, 1976; Kabsch, 1978), certaines procèdent par itérations successives (Sippl and Stegbuchner, 1991) ou par minimisation (McLachlan, 1982; McLachlan, 1979; Lesk, 1986).

La seconde mesure la plus connue, le $RMSD_d$, évite l'étape de superposition optimale car elle utilise les distances internes :

$$RMSD_d = \sqrt{\frac{\sum_{i=1}^{N-k} (d_{i,i+k}^A - d_{j,j+k}^B)^2}{N - k}} \quad (2.2)$$

où $d_{i,i+k}^A$ est la distance entre les atomes i et $i+k$ de la structure A et $d_{j,j+k}^B$ la distance entre les atomes correspondants j et $j+k$ de la structure B. La critique faite à cette mesure est qu'elle ne permet pas de différencier les images en miroir (Cohen and Sternberg, 1980), mais cette limitation n'est

pas critique dans les structures protéiques (Sippl, 1982). Les deux mesures sont somme toute aussi efficaces l'une que l'autre. Une relation les unissant a été proposée par M. J. Sippl (Sippl, 1982) : $RMSD_d = 0.75 \times RMSD_c + 0,19$.

Une autre mesure a été définie plus récemment (Chew et al., 1999; Kedem et al., 1999), qui a été nommée *URMS*. Tous les vecteurs définis par les C_α et le C_α suivant sont considérés comme unitaires car la distance entre deux C_α est à peu près fixe. Ils sont tous translatés à l'origine et c'est l'ensemble de ces vecteurs qui est superposé de manière optimale. La formule du *URMS* est la même que celle de $RMSD_c$.

D'autres RMSD ont été définis pour les angles (ϕ, ψ) :

$$RMSD_{\phi,\psi} = \sqrt{\frac{\sum_{i=1}^N (\phi_i^A - \phi_i^B)^2 + (\psi_i^A - \psi_i^B)^2}{N}} \quad (2.3)$$

et pour les angles (α, τ) :

$$RMSD_\alpha = \sqrt{\frac{\sum_{i=1}^{N-3} (\alpha_i^A - \alpha_i^B)^2}{N-3}} \quad (2.4)$$

2.3.2 Les coordonnées cartésiennes des C_α

Les premières méthodes de comparaison de structures protéiques utilisent la description la plus immédiate : les coordonnées cartésiennes. Elles sont généralement restreintes à la comparaison des C_α . Dans ces méthodes, qui sont plutôt des méthodes de comparaison globale, l'objectif est d'avoir le plus de C_α en correspondance avec le $RMSD_c$ le plus faible possible. Comme les C_α sont représentés par leurs coordonnées cartésiennes, ils sont considérés comme des points dans l'espace. La correspondance finale n'est donc pas obligatoirement un alignement mais en réalité c'est le plus souvent le cas à cause, non pas de la représentation, mais de l'algorithme utilisé. Comme toutes les méthodes décrites dans cette partie utilisent les coordonnées cartésiennes, toutes les distances entre C_α seront les distances entre les C_α en correspondance dans les deux protéines (et non les distances internes entre C_α d'une même protéine).

1) Les méthodes itératives de superposition-alignement

La méthode générale

Méthode générale de comparaison.

La méthode la plus utilisée pour trouver la meilleure correspondance $P(A, B)$ se déroule en deux étapes (voir aussi figure 2.3) :

Étape 1 trouver la superposition optimale des deux ensembles de points de $P(A,B)$ (transformation rigide, calcul de la matrice de translation-rotation),

Étape 2 déterminer une nouvelle et meilleure correspondance en prenant en compte la superposition.

Ces deux étapes sont répétées jusqu'à convergence (ou pendant un nombre limité d'itérations). Il faut bien sûr une correspondance initiale.

Les applications de cette méthode diffèrent par la correspondance initiale, la méthode de superposition optimale et la méthode de définition de la nouvelle correspondance. La superposition optimale des deux structures est trouvée avec l'une des méthodes décrites dans la partie "Les mesures de similarité : les RMSD" page 32.

L'alignement initial

La correspondance finale (alignement final) peut être très différent selon la correspondance initiale (alignement initial) (Feng and Sippl, 1996). Plusieurs méthodes ont été décrites pour la trouver. La plus utilisée est de rechercher d'abord de petits fragments similaires entre les deux protéines (*i.e.* ayant un $RMSD_c$ faible) et de n'initier l'algorithme qu'avec les meilleurs d'entre eux. Ces petits fragments similaires sont composés de 4 à 13 résidus contigus et sont souvent nommés **AFP**, pour *Aligned Fragment Pairs*. Plusieurs correspondances ou alignements finaux sont obtenus et seul le meilleur est conservé. Il est aussi possible de trouver l'alignement initial en alignant les séquences protéiques (Ogata et al., 1998) et il peut aussi être tiré au hasard (Subbiah et al., 1993).

Les deux méthodes principales utilisées lors de la seconde étape sont la mise en correspondance des C_α proches après superposition et la programmation dynamique.

Mise en correspondance des C_α proches

La première application de cette méthode semble être le premier programme de comparaison de structures (Rao and Rossmann, 1973; Rossmann and Argos, 1975). Ce programme, HOMOLOGY, est toujours disponible à l'adresse http://bilbo.bio.purdue.edu/~viruswww/Rossmann_home/softwares.shtml. La correspondance initiale est donnée par l'utilisateur et les distances entre les C_α après superposition optimale sont utilisées pour déterminer la nouvelle correspondance (calcul probabiliste). Ce programme peut aligner uniquement les C_α ou bien d'autres types d'atomes et il est le précurseur de beaucoup d'autres programmes alignant les C_α . Il est efficace pour aligner des structures proches mais l'est beaucoup moins pour des structures plus éloignées. Une autre limitation est la définition de la correspondance initiale.

Le programme MaxSub (Siew et al., 2000) a plus pour objectif de calculer un score de similarité entre deux structures (une structure réelle et une structure prédictive) que de chercher un alignement optimal de deux structures mais il utilise une version plus simple de la méthode : les C_α qui ont une distance inférieure à un seuil sont mis en correspondance. Les alignements initiaux sont toutes les

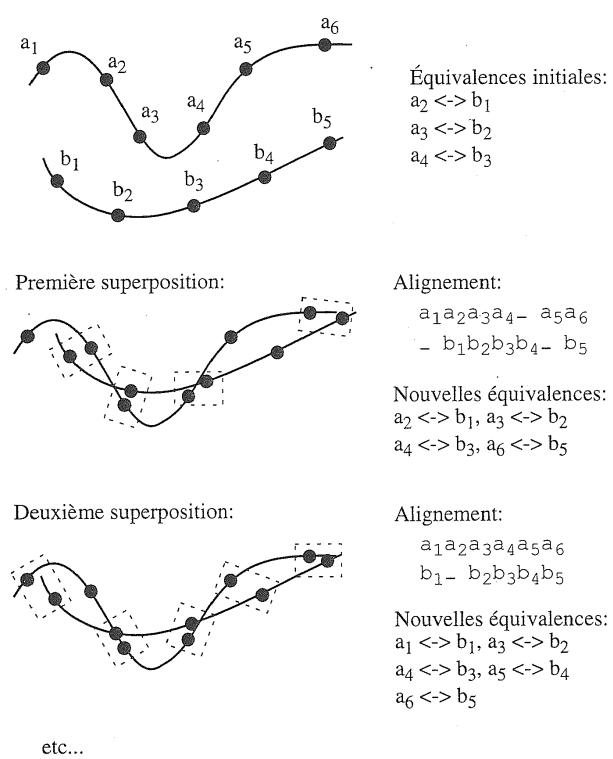


FIG. 2.3 – Figure extraite de la thèse de Vincent Escalier (Escalier, 1997). Schéma général des méthodes itératives de superposition-alignement. Une correspondance initiale est utilisée pour superposer les deux structures. On réalise ensuite un nouvel alignement qui permet d'obtenir une nouvelle correspondance. Cette correspondance permet de superposer à nouveau les structures et de réitérer le processus.

AFP de 4 résidus ayant un bon $RMSD_c$. Le score final est :

$$S = \frac{\sum_i \frac{1}{1 + (\frac{d_i}{d})^2}}{q} \quad (2.5)$$

où la somme est faite pour chaque paire de résidus i , q est le nombre de C_α dans la structure prédite, d_i est la distance entre les C_α en correspondance de la paire i (après superposition optimale) et d le seuil de distance. Ce score ne prend pas en compte les discontinuités et il est plutôt adapté à la comparaison de structures proches (comme une structure et son modèle prédit) à cause de la méthode d'alignement qui est plutôt globale.

Cette même version de la méthode est utilisée dans 3D-Hit (Plewny et al., 2002; Plewny et al., 2004). Ici aussi les alignements initiaux sont des AFP mais ils sont assez longs (13 résidus) et ils doivent satisfaire plus de conditions ($RMSD_c < 3\text{\AA}$, acide aminé central identique...). La différence entre 3D-Hit et MaxSub est qu'ici pour une AFP sélectionnée, seuls les 99 résidus autour de celle-ci sont superposés. Ensuite les paires de C_α qui sont à moins de 5\AA sont comptées et si l'on a plus de 35, les deux structures sont entièrement superposées et alignées comme dans MaxSub. L'étape intermédiaire de superposition et d'alignement permet d'éviter d'initier inutilement des alignements

et donc de gagner du temps. En effet, l'objectif de ce programme est de permettre la recherche de similarités structurales entre une structure protéique dite **requête** et toutes les structures d'une banque, ce qui est assez rare au niveau du squelette peptidique. Pour accélérer encore le programme, toutes les structures connues ont été découpées en fragments chevauchant de 13 résidus. Ces fragments ont été comparés entre eux et regroupés selon leur similarité structurale. Il y a un fragment représentatif pour chaque groupe. Pour retrouver rapidement à quelles structures un fragment représentatif correspond, les correspondances sont organisées dans une table de hachage. Lors de la recherche des AFP, il est alors très rapide de retrouver toutes les structures partageant un fragment avec la structure requête.

Le même genre de méthode a été développé récemment dans le programme LGA (Zemla, 2003) et dans le programme SAL (Kihara and Skolnick, 2003).

Alignement par programmation dynamique (étape 2)

L'algorithme le plus utilisé pour la seconde étape provient de l'alignement de séquences biologiques : il s'agit de la programmation dynamique (Needleman and Wunsch, 1970) nommée **NWS**. Cette technique a été utilisée notamment dans deux programmes connus : Structal (Subbiah et al., 1993; Levitt and Gerstein, 1998) et Prosup (Lackner et al., 2000; Feng and Sippl, 1996) mais aussi par d'autres (Barton and Sternberg, 1988; Cohen, 1997; Petitjean, 1998; May and Johnson, 1994; Kleywegt, 1996; Ogata et al., 1998; Gerstein and Levitt, 1998) (voir en annexe page 197 une brève description de la méthode). Avec cette méthode, la séquentialité est obligatoirement conservée. Le modèle d'évolution sur lequel elle est fondée est simple : il ne prend en compte que les substitutions et les insertions/délétions (« **indels** ») et non les translocations, duplications, réversions. Il nécessite aussi l'indépendance des positions entre elles (et donc aussi celle des substitutions et indels). Cette méthode permet de trouver l'alignement optimal entre deux séquences en s'appuyant sur les sous-alignements optimaux. Elle implique le calcul d'un score prenant en compte les substitutions et pénalisant les discontinuités.

Pour adapter cette méthode aux structures, le score de « substitution » a été remplacé par un score fondé sur le $RMSD_c$. Par exemple, dans le programme nommé Structal², le score défini par M. Levitt et M. Gerstein (Gerstein and Levitt, 1998) a pour formule :

$$w(a_i, b_j) = \frac{M}{\left(1 - \frac{d_{a_i, b_j}}{d_0}\right)^2} \quad (2.6)$$

avec $w(a_i, b_j)$ le coût de la « substitution » de a_i par b_j , $M = 20$ et $d_0 = 5\text{\AA}$. Les gap sont permis et pénalisés. Le score final a donc pour formule :

$$S = M \left(\sum_{i,j} \frac{1}{1 + \left(\frac{d_{a_i, b_j}}{d_0}\right)^2} - \frac{N_{gap}}{2} \right) \quad (2.7)$$

²La méthode décrite par S. Subbiah, D. Laurens et M. Levitt (Subbiah et al., 1993) est très similaire mais le score est différent.

où i et j sont les paires de résidus en équivalence des deux protéines, N_{gap} le nombre de gap, d_0 et M des constantes comme précédemment. D'après les auteurs ce score suit une distribution des valeurs extrêmes tandis que le $RMSD_c$ suit une distribution de la forme $\rho(Z) = e^{Z^4}$ (Levitt and Gerstein, 1998). Cette méthode est inspirée de celle utilisée dans le programme ALIGN (Cohen, 1997). Enfin, dans le « post-traitement », l'alignement final peut être affiné en enlevant certains C_α qui sont par exemple trop éloignés.

La méthode SHEBA (Jung and Lee, 2000) consiste à définir alignement un initial par programmation dynamique en prenant en compte la séquence, les structures secondaires, l'accessibilité au solvant et la polarité de l'environnement. Les structures sont ensuite superposées selon la méthode de Kabsch (Kabsch, 1976) et un nouvel alignement des résidus est calculé par programmation dynamique, le score dépendant des distances externes. La procédure superposition-alignement est répétée jusqu'à convergence.

La méthode TMAlign (Zhang and Skolnick, 2005) est basée sur le « TM-score » (Zhang and Skolnick, 2004) qui est pratiquement identique au score de M. Levitt et M. Gerstein (équation 2.7 sans pénalité de gap et divisé par la longueur d'une des deux protéines). L'alignement initial peut être défini par alignement des structures secondaires. A partir de cet alignement, les structures sont superposées non pas en minimisant le $RMSD_c$ mais en maximisant le TM-score. La procédure est alors habituelle : superposition-alignement par programmation dynamique.

Utilisation des deux méthodes précédentes

Il est possible d'utiliser conjointement les deux méthodes comme cela a été fait dans Prosup (Lackner et al., 2000; Feng and Sippl, 1996). La recherche de l'alignement initial consiste déjà à répéter plusieurs itérations de superposition-alignement comme dans 3D-Hit. Le nouvel alignement est augmenté en ajoutant toutes les paires de C_α dont la distance est inférieure à un seuil d_c . C'est seulement quand cette première étape a convergé qu'une seconde étape d'itérations superposition-alignement utilisant la méthode NWS a lieu. Le score utilisé est :

$$S_{ij} = \begin{cases} d_c^2 - d_{a_i, a_j}^2 & \text{si } d_{a_i, a_j} \leq d_c \\ 0 & \text{sinon} \end{cases} \quad (2.8)$$

Cette seconde étape permet donc d'ajouter à l'alignement des résidus qui ont été considérés comme trop éloignés lors de la première étape.

Une méthode originale MINAREA (Falicov and Cohen, 1996)

Les auteurs disent s'être inspiré des travaux Schulz (Schulz, 1977; Schulz, 1980). Le principe est de minimiser la surface entre les deux courbes en faisant des translation-rotations rigides (à partir des 3 angles d'Euler).

L'alignement initial est recherché en calculant le RMSD pour chaque superposition totale (de toute la longueur de la plus petite des protéines) possible de la petite protéine sur la grande protéine.

Pour les protéines, la surface entre les deux courbes dessinées par le squelette carboné est trian-

gulée c'est-à-dire, découpée en triangles dont deux sommets sont deux C_α d'une des protéines et le dernier sommet est sur l'autre protéine. Ils construisent une matrice à partir de ces triangles et trouvent le chemin donnant la surface minimale par programmation dynamique (NWS). Ayant trouvé la série de triangles, ils minimisent la surface par translation-rotation en utilisant l'une des 3 méthodes d'optimisation suivantes : méthode de Powell³, du simplexe⁴ ou méthode des gradients conjugués⁵. Ils calculent ensuite un score : $FC = \frac{AF}{\sum_{i,j} \|C_\alpha^A - C_\alpha^B\|}$ où AF est la surface minimale (qui est donc divisée par la somme des distances entre toutes les paires possibles de C_α des 2 protéines).

2) Méthodes basées sur des fragments structuraux similaires

Principe de ce type de méthodes

Les méthodes décrites précédemment ne sont pas très efficaces pour comparer des structures éloignées à cause du caractère global de l'alignement. Pour palier cette limitation en utilisant le même niveau de représentation (les C_α) et le même type de score (fondé sur le $RMSD_c$), Remington et Matthews ont calculé les $RMSD_c$ entre tous les fragments d'une taille donnée (au minimum 20 résidus) de deux structures et ont étudié leur distribution (Remington and Matthews, 1978; Remington and Matthews, 1980). Ils ont ainsi pu caractériser les paires de protéines similaires mais ils ne définissent pas d'alignement des deux structures. Les méthodes décrites ci-après découlent de ces premiers travaux.

Méthode générale de comparaison.

Elles se déroulent en trois étapes :

Étape 1 trouver tous les petits fragments similaires dans les deux structures (AFP), le plus souvent des paires de fragments structuraux dont le $RMSD_c$ est faible ;

Étape 2 trouver la ou les meilleures séries d'AFP (assemblage) ;

Étape 3 affiner l'alignement ou la correspondance au niveau des résidus ; une méthode itérative de superposition-alignement peut être utilisée.

Elles sont similaires à beaucoup de méthodes itératives de superposition-alignement dans leur première étape mais la seconde phase diffère largement selon les programmes.

Assemblage par programmation dynamique

Certains programmes utilisent la programmation dynamique pour trouver la meilleure série d'AFP (Zuker and Somorjai, 1989). Dans le programme MAMMOTH (Ortiz et al., 2002), des AFP de 7 résidus sont recherchées en calculant non pas le $RMSD_c$ mais le $URMS$ (défini page 33) (Chew et al., 1999; Kedem et al., 1999). Pour toutes les AFP possibles, un score de similitude est calculé selon la formule $S = \frac{URMS^{expected} - URMS}{URMS^{expected}} \times 10$ où $URMS^{expected}$ est l'URMS minimal attendu entre deux

³succession de minimisations monodimensionnelles pour chaque paramètre, itérée jusqu'à convergence

⁴succession de minimisations dans la direction du meilleur point trouvé

⁵succession de minimisations selon des directions données par les gradients conjugués par rapport au Hessien

ensembles aléatoires de vecteurs ($URMS^{expected} = \sqrt{2 - \frac{2,84}{\sqrt{n}}}$ avec n le nombre de vecteurs) ; le score est mis à 0 s'il est négatif. La technique permet alors de trouver le meilleur alignement local (comme dans l'alignement de séquences de type Smith et Waterman (Smith and Waterman, 1981)). Le score final suit une distribution des valeurs extrêmes (Karlin and Altschul, 1990) et une *p-value* est calculée pour l'alignement final (voir annexes page 198).

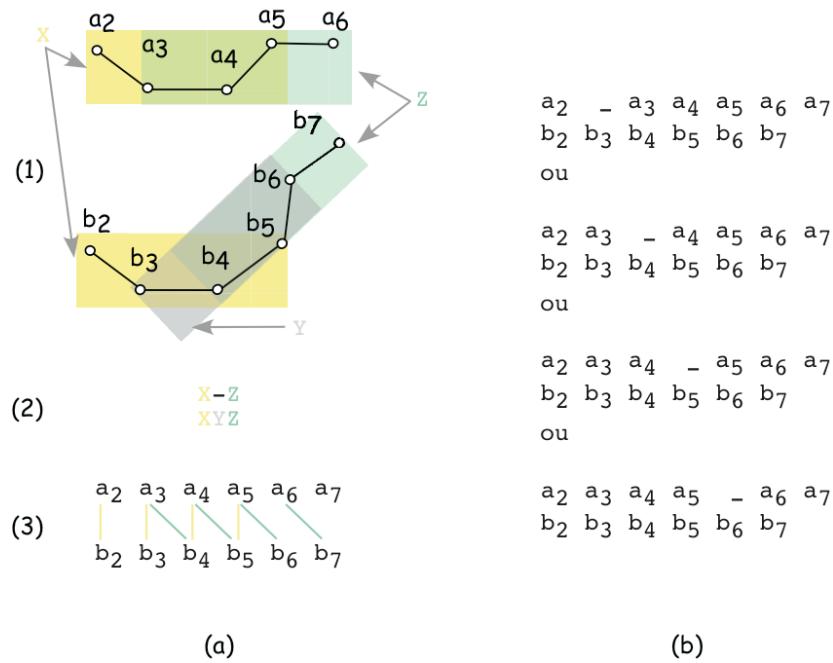


FIG. 2.4 – Exemple de blocs structuraux similaires chevauchant avec gap.

- (a.1) Représentation schématique des deux structures. Les blocs structuraux (AFP) ont 4 résidus. Les deux premiers blocs structuraux jaunes nommés X sont similaires dans les deux structures mais le bloc gris Y de la structure B n'a pas d'équivalent dans la structure A. Par contre les blocs structuraux verts Z sont similaires.
 (a.2) Si seuls les blocs sont alignés, un gap est inséré en face du bloc Y.
 (a.3) Au niveau des résidus, comme les correspondances ne sont pas les mêmes dans les deux blocs, certains résidus de la protéine sont en correspondance avec deux résidus de la protéine B.
 (b) Plusieurs alignement des séquences protéiques sont alors possibles, dans le cas présent 4. Comme en général dans une méthode de comparaison structurale le bloc est assigné à un résidu, si un bloc ne trouve pas de correspondant, c'est devant le résidu que se trouve le gap, mais la position du gap est un choix arbitraire.

L'utilisation de la programmation dynamique pour trouver la meilleure série d'AFP pose plus de problèmes théoriques d'application que lors de son utilisation dans les méthodes itératives de superposition-alignement d'atomes. En effet, comme il est montré dans la figure 2.4, il existe plusieurs alignements possibles au niveau des résidus lorsqu'un gap est présent dans l'alignement des AFP. L'alignement par programmation dynamique doit donc être fait avec les blocs eux mêmes et non avec les résidus. En effet, chaque score $S_{i,j}$, i et j étant des indices de résidus, implique en réalité dans la méthode précédente 7 paires de résidus consécutifs. Si un gap est inséré à un endroit de l'alignement, les scores de toutes les AFP précédentes alignées couvrant le résidu dorénavant sans correspondant sont faux car ces AFP impliquent une correspondance qui n'est plus dans l'alignement. Par

contre si l'alignement est fait au niveau de blocs le score calculé est juste, la similarité entre les blocs étant toujours réelle. Le retour à l'alignement des résidus pose quand même le problème du choix de l'emplacement du gap dans la séquence. Evidemment, le problème de l'indépendance des positions dans les structures, principe de base pour appliquer la programmation dynamique, reste entier (par ricochet, il pose aussi celui de l'indépendance des positions dans les alignement de séquences...).

Le programme WHAT IF

La méthode implémentée dans le programme WHATIF (Vriend, 1990; Vriend and Sander, 1991) cherche des AFP de taille fixe (10 à 15 résidus), mais le $RMSD_c$ n'est calculé que si les distances internes entre le premier C_α et les 5 derniers C_α sont les mêmes dans les deux fragments. Si le $RMSD_c$ est assez faible, l'AFP est étendue aux C_α voisins tant que le $RMSD_c$ reste en dessous du seuil. Les ensembles d'AFP sont ensuite générés de la manière suivante : les deux structures sont superposées selon l'ensemble d'AFP courant et une AFP n'est ajoutée que si les centres de gravité de ses deux fragments ne sont pas trop éloignés. La séquentialité des AFP n'est pas toujours observée, donc c'est une correspondance qui est finalement obtenue. La superposition globale des deux structures est alors recalculée pour cet ensemble d'AFP en prenant en compte les nouvelles correspondances entre C_α . Finalement, seule la correspondance la plus longue est conservée et affinée.

Adaptation de l'algorithme « classification hiérarchique »

Le squelette peptidique est décrit par les atomes (N,C α ,C,O) dans la méthode de Wodak et coll. (Bouillonnet et al., 1995; Ochagavia et al., 2002). La méthode d'alignement se déroule en 3 étapes : rechercher les AFP, trouver la meilleure série d'AFP, affiner l'alignement.

Les AFP (5 résidus) sont recherchées en calculant le $RMSD_c$ entre tous les fragments chevauchants mais seulement s'ils respectent certaines conditions (distances entre les premiers et derniers résidus proches, positions respectives des AFP dans les séquences pas trop différentes...).

Les séries d'AFP sont mises en évidence avec un algorithme de classification hiérarchique sur liens multiples sous contrainte (*constrained multiple linkage tree algorithm*). Au départ, chaque AFP est considérée comme une seule série. L'algorithme essaye de joindre toutes les paires de séries formées à une étape donnée. Toutes les paires de séries compatibles, c'est-à-dire non chevauchantes, sont combinées et le $RMSD_c$ des résidus en équivalence dans la paire est calculé. La paire de meilleur $RMSD_c$ est conservée et forme une nouvelle série. Il existe des contraintes à la fusion de deux séries (contraintes de séquentialité, contraintes sur le RMSD...). Ces assemblages se font le long d'un arbre (voir figure 2.5).

Les alignements obtenus sont affinés en ajoutant les résidus du côté N ou C-terminal des segments continus si le RMSD reste adéquat. Finalement, comme plusieurs séries d'AFP donnent le même alignement final, celui-ci est redécomposé en chacune de ses séries (*i.e.* les branches de l'arbre qui mènent à cet alignement), le RSMD de la série est calculé à chaque étape. La série dont le RMSD à chaque étape varie le moins est gardée. C'est donc celle ayant le plus d'AFP car le $RMSD_c$ final est le même.

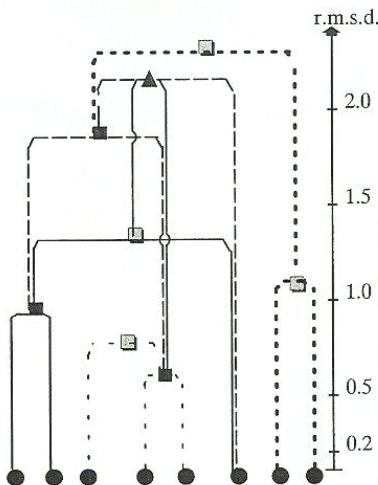


FIG. 2.5 – Figure extraite de l’article de Boutonnet et coll. (Boutonnet et al., 1995). Représentation schématique de l’arbre de classification hiérarchique sur liens multiples. Toutes les paires de segments de 5 résidus ayant un $RMSD_c$ acceptable sont les suites initiales (ronds noirs). Les séries provenant de la réunion de deux séries sont représentées par des carrés ou des triangles. Les carrés ou triangles gris sont des séries qui ne sont combinées qu’une seule fois avec une autre série tandis que les noirs sont des séries combinées au moins deux fois. Les nœuds (qui représentent donc des séries) en forme de triangle peuvent être atteints via des chemins différents de l’arbre. Les $RMSD_c$ des séries sont représentés par l’échelle à droite. La position d’un nœud dans l’arbre dépend donc du $RMSD_c$ de la série mais comme cette valeur croît avec le nombre de résidus en équivalence, l’ordre des combinaisons est aussi respecté. Pour plus de simplicité toutes les paires initiales de segments de 5 résidus ont été mises au même niveau même si leur $RMSD_c$ varie.

Méthodes d’assemblage « flexible »

Certaines méthodes ont essayé de prendre en compte la flexibilité des protéines lors de l’étape d’assemblage (Wriggers and Schulten, 1997; Boutonnet et al., 1995; Ochagavia et al., 2002; Shatsky et al., 2002; Shatsky et al., 2004). L’idée générale est que l’alignement de deux protéines se décompose en plusieurs segments rigides joints par des zones flexibles. Cette idée est aussi présente dans la dernière partie de la méthode développée Wodak (Boutonnet et al., 1995; Ochagavia et al., 2002).

Flexprot (Shatsky et al., 2002; Shatsky et al., 2004) Dans le programme Flexprot (Shatsky et al., 2002; Shatsky et al., 2004), les AFP sont donc déterminées par rapport au $RMSD_c$ et quelques étapes itératives de superposition-ajout des C_α contigus sont effectuées pour les étendre au maximum (le $RMSD_c$ doit rester inférieur à 3Å). La méthode d’assemblage est ensuite inspirée de celle utilisée dans FASTA (Pearson and Lipman, 1988), elle nécessite la construction d’un graphe orienté acyclique. Toutes les AFP sont des nœuds du graphe et un arc (une arête orientée) est défini entre deux AFP si celles-ci peuvent se succéder dans l’alignement final. La séquentialité est donc conservée. Un poids est ensuite associé aux arcs, il pénalise les gaps. Les chemins possibles sont ensuite recherchés dans ce graphe. Finalement, les alignements disjoints sont regroupés en un seul alignement lorsque c’est possible.

L’assemblage dit flexible est aussi utilisé par Wriggers et coll. (Wriggers and Schulten, 1997). La méthode pour trouver les fragments structuraux similaires aux deux protéines s’appuie sur celle

décrise par A. Lesk (Lesk, 1979; Lesk, 1991) qui classe les atomes par leur conformité avec la structure superposée, c'est-à-dire élimine progressivement les résidus qui coïncident mal lors de la superposition. La méthode de Wrigger et coll. est surtout utilisée pour trouver les zones rigides sur les conformations d'une dynamique moléculaire par exemple. Wriggers et coll. partent itérativement de petits sous-ensembles cohérents et leur ajoutent les atomes qui respectent une distance seuil lors de la superposition du sous-ensemble considéré, jusqu'à convergence. La procédure s'arrête lorsque la (ou les) protéine(s) est (sont) complètement partitionnée(s) en sous-structures s'ajustant bien deux à deux. Deux modes sont possibles, l'un - lent - maintient la connectivité des segments, l'autre - rapide - ne la maintient pas.

2.3.3 Les coordonnées internes : les distances « internes »

Pour éviter l'étape de superposition, nous avons vu qu'il faut utiliser les coordonnées internes. Les méthodes décrites dans cette partie comparent les structures au niveau peptique en ne prenant en compte que les C_α ou quelques autres atomes. Pour décrire ces atomes, les distances dites internes, c'est-à-dire entre les atomes d'une même structure, sont calculées. Il y a alors $\frac{N^2}{2}$ descripteurs pour chaque structure, N étant le nombre d'atomes, au lieu des $3 \times N$ paramètres de la description précédente en coordonnées cartésiennes. Ces $\frac{N^2}{2}$ descripteurs sont souvent présentés sous la forme d'une matrice (symétrique) dite de distances internes. Les matrices de contact sont aussi parfois utilisées : elles contiennent 1 si la paire d'atomes satisfait à certaines conditions (par exemple si la distance interne est en dessous d'un seuil) et 0 sinon.

Bien que les premières matrices de distances aient fait leur apparition très tôt (Phillips, 1970), les premiers essais de comparaison de structures les utilisant n'ont pas été très fructueux (Nishikawa and Ooi, 1974; Padlan and Davies, 1975; Lieberman, 1982). La méthode est en effet coûteuse en ressources et ne s'applique qu'aux structures proches. Comme pour les méthodes précédentes, des mesures de similarités ont été définies en comparant par exemple toutes les distances internes entre C_α sur une même diagonale, c'est-à-dire correspondant à un même décalage (Sippl, 1982). Dans les années 90, les premiers programmes utilisant les distances internes sont apparus. Comme il est impossible de tester toutes les correspondances possibles, diverses heuristiques ont été utilisées.

Les différentes méthodes développées se classent en trois grandes catégories : méthodes utilisant la programmation dynamique, méthode d'assemblage d'AFP, méthodes utilisant les graphes. Des méthodes de calcul de scores de similarité fondées sur les distances internes ont aussi été développées, elles seront abordées à la fin de cette section.

1) Méthodes utilisant la programmation dynamique

En réalité, seul le programme SSAP ou ses dérivés utilisent la programmation dynamique et les distances internes. En effet il n'y a pas plusieurs manières d'utiliser la programmation dynamique et les matrices de distances internes : la seule est la méthode dite de double programmation dynamique

(DDP). Le score étant basé sur les distances internes, il faut faire toutes les soustractions possibles des deux demi-matrices, c'est-à-dire en se décalant vers le bas et en se décalant vers la droite. Le programme SSAP utilise à la fois les distances internes et d'autres descripteurs tels que l'information de séquence, l'accessibilité au solvant *etc.*, et il est donc décrit dans la section « Méthodes utilisant plusieurs descriptions ».

2) Méthodes d'assemblage de fragments similaires

Principe de cette catégorie de méthodes

Le principe est de rechercher de petits fragments similaires (AFP), puis la meilleure série d'AFP. Les deux programmes les plus connus, DALI et CE utilisent ce principe. Plus récent, FATCAT l'utilise aussi. L'originalité de chaque méthode se situe plus au niveau de la méthode d'assemblage que de la recherche des AFP.

DALI

Dans cette méthode (Holm and Sander, 1993), les deux matrices de distances internes sont divisées en sous-matrices de taille fixe (6). Il y a donc N^2 « hexa-matrices » pour chaque structure, N étant la longueur d'une protéine (les matrices ne sont composées que de C_α contigus). Certaines hexa-matrices représentant des SSE sont modifiées ou filtrées (par exemple, pour réduire leur nombre, les hexa-matrices chevauchantes représentant des SSE sont combinées pour former des matrices de taille supérieure et pour un SSE, seule est conservée la matrice ayant la distance moyenne la plus petite). Ensuite, il faut trouver les hexa-matrices similaires dans les deux protéines. Pour éviter de les comparer toutes, certains filtres sont utilisés. Pour les paires d'hexa-matrices restantes, un score élastique est calculé :

$$S = \sum_{i=1}^L \sum_{j=1}^L \phi(i, j) \quad (2.9)$$

où i et j sont les indices des C_α dans la matrice (et non leur indice dans chaque protéine) et ϕ la fonction de mesure de similarité.

Pour cette mesure de similarité, les auteurs introduisent une fonction dite élastique :

$$\phi^E(i, j) = \begin{cases} \left(\Theta_E - \frac{|d_{i,j}^A - d_{i,j}^B|}{d_{i,j}^*} \right) w(d_{i,j}^*) & \text{si } i \neq j \\ \Theta_E & \text{si } i = j \end{cases} \quad (2.10)$$

où $d_{i,j}^*$ est la moyenne de $d_{i,j}^A$ et $d_{i,j}^B$, Θ_E une constante égale à 0,2 (qui permet donc 20% de variation) et $w(r)$ une fonction d'enveloppe $e^{-\frac{r^2}{\alpha^2}}$ avec $\alpha = 20\text{\AA}$. $w(r)$ permet de moins prendre en compte les différences entre grandes distances (20\AA est la taille typique d'un domaine).

Cette fonction est plus tolérante pour l'accumulation de différences tout au long de l'alignement, elle permet donc de maximiser la longueur de l'alignement en même temps qu'elle minimise les

différences.

Les 40 000 meilleures paires d'hexa-matrices sont conservées. Quand deux hexa-matrices sont considérées comme similaires, les résidus correspondant des deux protéines sont alignés. Par exemple, si dans l'hexa-matrice de la protéine A, il y a les distances entre les C_α 3 à 8 et les C_α 20 à 25 et que l'hexa-matrice de la protéine B contient les distances entre les C_α 12 à 17 et les C_α 40 à 45, les résidus 3 à 8 de la protéine A sont alignés avec les résidus 12 à 17 de la protéine B et les résidus 20 à 25 de la protéine A avec les résidus 40 à 45 de la protéine B. Il faut noter que les deux paires de fragments peuvent donc être disjointes, et les hexa-matrices ne sont donc pas des AFP - qui sont elles définies comme des successions de résidus contigus. Je n'utiliserai donc pas ce terme par la suite.

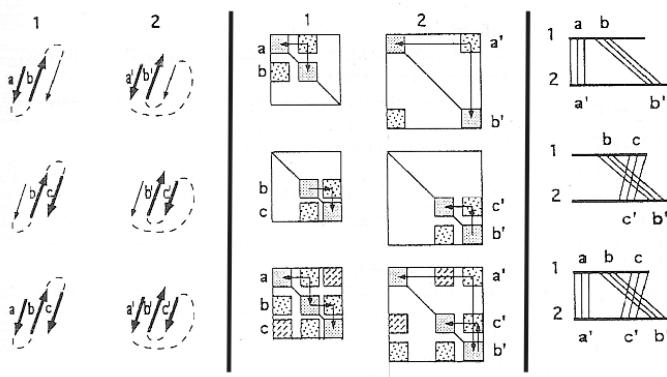


FIG. 2.6 – Figure extraite de (Holm and Sander, 1993). L'alignement (en cours) des deux structures est représenté de trois manières. A gauche : représentation très schématique de la structure des chaînes peptidiques des protéines 1 et 2. Trois fragments structuraux de 6 résidus sont représentés par des flèches dans chaque structure. Ils sont reliés de manière différente dans les deux structures. Au milieu : les deux matrices de distances internes des protéines 1 et 2. Les petits carrés grisés représentant des hexa-matrices. A droite : les alignement structuraux projetés sur les séquences protéiques.

Première Ligne : Les distances internes sont calculées entre les fragments *a* et *b* de structures de la protéine 1 et les fragments *a'* et *b'* de la structure 2. Il y a donc quatre hexa-matrices pour chaque protéine (par exemple *aa*, *ab*, *ba*, *bb* pour la protéine 1). Par exemple pour la structure 1, l'hexa-matrice *aa* contient toutes les distances entre les résidus du fragment *a*, l'hexa-matrice *ab* contient toutes les distances entre les résidus des fragments *a* et *b*. Comme les matrices de distances sont symétriques, l'hexa-matrice *ab* contient les mêmes valeurs que l'hexa-matrice *ba*. Il a été déterminé à l'étape précédente que *a* est similaire à *a'* et *b* à *b'*. Le question est maintenant de savoir si le fragment *ab* est similaire à *a'b'*, i.e. si les fragments peuvent être assemblés. Pour cela, le score *S* est calculé avec les matrices *ab* et *a'b'* (ou leurs symétriques).

Deuxième ligne : comme à la ligne précédente mais les fragments *b* et *c* ne sont pas dans le même ordre que les fragments *b'* et *c'* (visible dans les schémas de la première colonne et de la troisième colonne).

Troisième ligne : combinaison des deux assemblages précédents.

Ensuite commence la seconde étape d'assemblage. et comme il est impossible de rechercher exhaustivement toutes les combinaisons d'hexa-matrices, l'algorithme de Monte Carlo est utilisé. Le principe de l'optimisation par Monte Carlo est d'explorer aléatoirement l'espace de recherche pour améliorer l'alignement. Les paires d'hexa-matrices subsistantes sont subdivisées chacune en 3 tétra-matrices. L'objectif est donc de trouver la meilleure série de tétra-matrices, chevauchantes ou non, dans le même ordre ou non, dans les deux protéines (voir figure 2.6). Finalement, c'est donc une correspondance et non un alignement qui est obtenu. La méthode se décompose en deux types d'étapes : des étapes dites d'« élagage » (*trimming*) où les tétra-matrices qui font chuter le score de la série sont éliminées et des étapes dites d'« expansion » où l'algorithme de Monte Carlo est utilisé. Lors d'une

étape de l’algorithme d’expansion, une tétra-matrice est tirée au hasard et elle est ajoutée à la série avec une probabilité $p = \exp(B * (S' - S))$ où S' est le score en incluant la nouvelle tétra-matrice, S l’ancien score et B une constante. Comme dans le recuit simulé, cette constante varie au long de la procédure : lorsqu’un haut score est atteint, elle est diminuée pour permettre d’affiner l’alignement ; lorsque le score est faible elle est augmentée pour pouvoir explorer l’espace des alignements possibles. Comme les tétra-matrices représentent des alignements discontinus, la séquentialité de l’alignement n’est pas obligatoirement observée. Seules les tétra-matrices contenant au moins une des paires de résidus en correspondance peuvent être ajoutées (rappel : les résidus dans une tétra-matrice ne sont pas forcément contigus). Il est possible que certaines correspondances entre résidus soient invalidées par les nouvelles correspondances d’une nouvelle tétra-matrice (non située sur la même diagonale que la tétra-matrice qu’elle chevauche), elles sont alors interdites. Les étapes d’élagage ont lieu après une succession de 5 étapes d’expansion.

L’algorithme d’assemblage entier est initié avec plusieurs points de départ et la correspondance finale entre résidus est affinée en enlevant 30% des correspondances et en réitérant l’algorithme à partir de ces nouveaux points de départ. L’algorithme de Monte Carlo a été remplacé par un algorithme de *Branch and Bound* dans les versions suivantes de DALI.

Ce programme fait partie des meilleures méthodes de comparaison structurale. Cependant, la méthode de comparaison reste assez globale. Il a été utilisé pour classer les protéines (la classification est nommée FSSP (Holm and Sander, 1996b)) et il est mis à disposition sur un serveur (Holm and Sander, 1995b). Il est à noter que dans sa version serveur, DALI utilise en plus des pré-filtres basés sur les structures secondaires (Holm and Sander, 1995a), ce qui détériore la finesse de la recherche, puisque seules les structures de la base de données répondant aux critères de SSE de la structure requête sont examinées. On verra en effet plus loin qu’à part VAST (Gibrat et al., 1996), les méthodes fondées sur les structures secondaires montrent de nettement moins bonnes performances que les méthodes décrites ici.

CE

La méthode développée dans le programme CE (*Combinatorial Extension* (Shindyalov and Bourne, 1998)) utilise la plupart du temps les distances internes entre C_α .

La première étape de recherche de petits fragments similaires (ici 8 résidus) ne prend en compte que les distances internes : il faut que la moyenne des différences des distances internes soient inférieure à 3Å. Ces paires de fragments sont des AFP. L’étape d’assemblage consiste à ajouter une AFP à la série déjà construite si plusieurs conditions sont respectées :

- la nouvelle AFP n’est chevauchante avec aucune de celles déjà présentes.
- la nouvelle AFP est contiguë avec une des AFP de la série sur au moins une des deux protéines.
- s’il y a un gap, il doit être de longueur inférieure à 30 résidus.
- la distance moyenne entre toutes les paires d’AFP (nouvelle AFP et AFP de la série) doit être inférieure à 4Å.

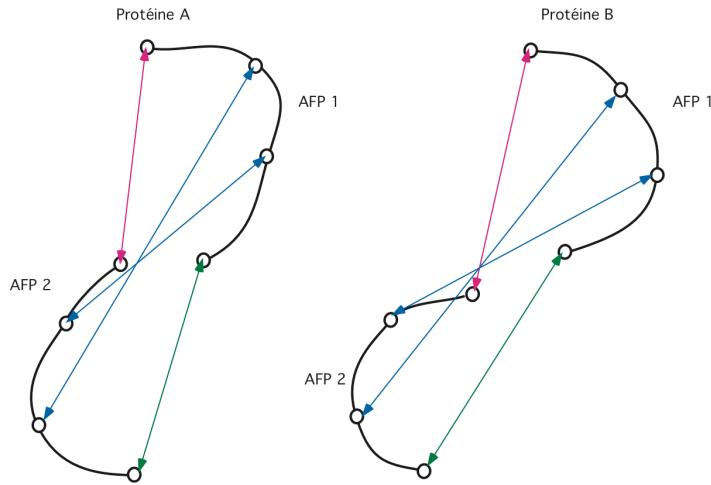


FIG. 2.7 – Les distances prises en compte dans le score de CE

$$D_{i,j} = \frac{1}{m} |d_{p_i^A, p_j^A}^A - d_{p_i^B, p_j^B}^B| + |d_{p_i^A+m-1, p_j^A+m-1}^A - d_{p_i^B+m-1, p_j^B+m-1}^B| + \sum_{k=1}^{m-2} |d_{p_i^A+m-1, j+m-1-k}^A - d_{p_i^B+m-1, j+m-1-k}^B|.$$

Pour simplifier, il n'y a que 4 résidus par AFP.

Elle est définie par

$$\begin{aligned} D_{i,j} = & \frac{1}{m} |d_{p_i^A, p_j^A}^A - d_{p_i^B, p_j^B}^B| \\ & + |d_{p_i^A+m-1, p_j^A+m-1}^A - d_{p_i^B+m-1, p_j^B+m-1}^B| \\ & + \sum_{k=1}^{m-2} |d_{p_i^A+m-1, p_j^A+m-1-k}^A - d_{p_i^B+m-1, p_j^B+m-1-k}^B| \end{aligned}$$

où p_i^A et p_j^A sont les indices des premiers résidus des AFP i et j dans la protéine A, p_i^B et p_j^B les indices des premiers résidus des AFP i et j dans la protéine B et m la longueur des AFP. C'est en fait la moyenne entre les différences de distances internes entre les deux premiers résidus des deux AFP, les deux derniers résidus des AFP et entre tous les autres résidus mis en correspondance (voir figure 2.7).

Si toutes ces conditions sont respectées pour plusieurs AFP, seul la meilleure est ajoutée - celle dont la moyenne des $D_{i,j}$ est la plus faible. Plusieurs alignements sont initiés avec toutes les AFP dont les distances internes sont en moyenne inférieure à 3 Å. Parmi les 20 meilleurs alignements finaux obtenus, seul celui ayant le meilleur $RMSD_c$ est conservé. Il est ensuite affiné notamment par une procédure classique de superposition-alignement par NWS. Un Z-score est aussi calculé pour cet alignement le plus long en évaluant la probabilité de trouver un alignement de même longueur avec autant ou moins de gaps et à une distance plus petite ou égale. Cette probabilité est évaluée à partir d'alignements obtenus par comparaisons aléatoires de structures d'un ensemble non redondant de la PDB (Hobohm et al., 1992) et calculés avec le même algorithme.

Méthodes d'assemblage « flexible »

FATCAT Les AFP sont ici des fragments de 8 résidus dont le $RMSD_c$ est inférieur à 3Å. FATCAT (Ye and Godzik, 2003; Ye and Godzik, 2004b; Ye and Godzik, 2004a) utilise donc à la fois les représentations en coordonnées cartésiennes et les distances internes. La méthode consiste à combiner les AFP en prenant en compte les gaps et les mésappariements. La meilleure série d'AFP est recherchée par programmation dynamique mais le système de score utilisé prend en compte ces gaps et mésappariements. Soit t le nombre maximal de mésappariements, $S(k)$ le meilleur score du chemin aboutissant à l'AFP k peut être calculé avec la formule :

$$S(k) = a(k) + \max_0^{\{S(m) + c(m \rightarrow k)\}} \text{ avec } T(k) \leq t \quad (2.11)$$

où $a(k)$ est le score de l'AFP k elle même qui dépend de son $RMSD(i)$, $S(m)$ est le score du chemin aboutissant en m (l'AFP précédent k) et $c(m \rightarrow k)$ est le coût pour introduire une connexion entre les AFP m et k : ce coût dépend du score entre les deux AFP m et k , des mésappariements et des gaps de la connexion. Une condition est ajoutée à la connexion de deux AFP : il faut que le nombre total de mésappariements $T(k)$ de l'alignement soit inférieur à un seuil t .

Un score final est calculé tel que $s = cs \times \sqrt{\frac{L}{RMSD \times N}}$ où cs est le score d'alignement, L le nombre de C_α alignés, N le nombre de blocs dans l'alignement. Comme ce score suit une distribution des valeurs extrêmes, une *p-value* peut être calculée.

3) Méthodes utilisant les graphes

Les distances internes peuvent être considérées comme des « relations » entre les atomes. La représentation de chaque structure par un graphe ayant le Ca pour sommets les atomes et des arêtes pondérées les distances semble donc évidente. Cependant, la comparaison de graphes n'est pas un problème trivial, ce type de méthodes est donc apparu tardivement, il a été utilisé par Barakat et Haneef (Barakat and Dean, 1991; Subbarao and Haneef, 1991). Dans cette dernière méthode, la contrainte de séquentialité a permis de réduire le problème de recherche des sous-graphes communs (graphes isomorphes). L'alignement est ensuite affiné en ajoutant les C_α voisins des régions communes.

Dans le programme récent FAST (Zhu and Weng, 2005), la méthode est un peu différente car les sommets du graphe sont des paires de C_α ($C_{\alpha_i}^A, C_{\alpha_j}^B$) et une arête est définie entre deux sommets quand la différence entre les distances internes est inférieure à un seuil ($|d_{i,k}^A - d_{j,l}^B| < \text{seuil}$). Les arêtes sont pondérées avec ces distances mais aussi avec la valeur de certains angles entre pseudo-liaisons définis sur les C_α , un score $e_{ij,kl}$ est ainsi obtenu pour chaque arête. Pour chaque sommet, la somme des scores de ses arêtes forme le score $T_{i,j}$ de la mise en correspondance du C_α i de la protéine A avec le C_α j de la protéine B. Ce score est utilisé pour l'étape finale d'alignement par programmation dynamique (NWS). Pour limiter le nombre de sommets dans le graphe, plusieurs filtres sont utilisés, notamment

les distances internes entre C_α .

4) Méthodes sans alignement

Le calcul d'un score de similarité est un problème qui se pose fréquemment lors de l'étude des structures protéiques. Il se pose notamment lors de l'évaluation des modèles déterminés par les méthodes de prédition de structures à partir de la séquence protéique. Certains programmes sont donc dédiés à cette tache.

Il est difficile de mesurer à quel point deux structures se ressemblent car selon les avis la ressemblance doit être locale ou globale. Nous avons vu que le $RMSD_c$ est souvent utilisé. Cette mesure est globale et elle n'est donc pas appropriée aux structures de ressemblance lointaine. Comme on l'a déjà vu, chaque méthode a en général développé sa propre mesure, néanmoins certains programmes sont dédiés à la résolution de ce problème de mesure de similarité.

Le score CONGENEAL (Yee and Dill, 1993) est fondé sur les différences de distances internes entre C_α tandis que le score de PRIDE (Carugo and Pongor, 2002) est fondé sur la distribution des distances internes $d(C_\alpha i, C_\alpha i + k)$. Ce sont des scores plutôt globaux.

La mesure de dissimilarité de CONGENEAL entre deux structures est :

$$d(A, B) = \frac{\sum_{i=1}^N \sum_{j=i+2}^N |b_{i,j}^p - a_{i,j}^{-p}|}{\frac{1}{2}(\sum_{i=1}^N \sum_{j=i+2}^N b_{i,j}^p + \sum_{i=1}^N \sum_{j=i+2}^N a_{i,j}^p)} \quad (2.12)$$

où $a_{i,j}$ la distance entre les carbones i et j de la protéine A, *idem* pour $b_{i,j}$ dans la protéine B, p une constante négative.

Ce score revient à faire la somme des différences sur les cartes de distances - élevées à la puissance $-p$ - des deux protéines. Ici les deux protéines sont de longueur N . Si leurs longueurs sont différentes, cette mesure est calculée pour tous les décalages possibles et le minimum - meilleure similarité - est conservé (le calcul est en fait légèrement plus complexe). Ce score est rapide à calculer et les auteurs s'en sont servi pour classer la PDB en familles.

Dans PRIDE, Carugo et Pongor (Carugo and Pongor, 2002) calculent la probabilité que deux structures soient similaires en comparant simplement les distributions des distances internes $d(C_\alpha i, C_\alpha i + k)$ des deux protéines en utilisant les tableaux de contingence et le χ^2 .

Dans la méthodes SGM, une le squelette peptidique est modélisée par un courbe mathématique en trois dimensions. Cette courbe est ensuite décrite par 30 valeurs (nœuds invariants ou invariants de Vassiliev) qui sont comparées et permettent de calculer un score de similarité nommé *scaled Gauss metric* ou SGM.

2.3.4 Les repères internes

Méthodes dérivées des algorithmes pour la reconnaissance de formes

Il existe bien sûr des méthodes travaillant au niveau du squelette peptidique et dans l'espace cartésien ne rentrant dans aucune des deux catégories précédentes. Deux de ces méthodes ont adapté un algorithme développé pour la reconnaissance de formes - le hachage géométrique (*Geometric Hashing*). Des repères sont définis par certains atomes ((N, C_α, C) d'un même résidu (Pennec and Ayache, 1994; Pennec and Ayache, 1998) ou des triplets de C_α (Nussinov and Wolfson, 1991)). Les positions des autres atomes sont calculées dans ces repères et sont rangées dans une table de hachage. Pour comparer une structure à celle stockée dans la table, il suffit de calculer les paires (repères, listes de points) dans cette structure et de les mettre en équivalence avec celles de la table. Il s'agit en fait d'un système de « votes » pour les repères de la structure de référence. Les 2 structures sont superposées à l'aide du repère qui dépasse un seuil de votes, en minimisant le RMSD sur les C_α ayant voté pour ce repère.

Une description symbolique proche des angles internes (Matsuda et al., 1997)

La structure du squelette est ici codée en une suite de symboles, l'alphabet se composant 20 symboles. L'attribution de ces symboles est faite comme suit : un repère orthonormé est défini pour chaque triplet de C_α contigus ($i-2, i-1, i$) et un icosaèdre (20 faces) est placé dans ce repère tel que son centre de gravité soit en C_{α_i} . Un symbole est attribué à chaque face de l'icosaèdre et le symbole de la face la plus proche du $C_{\alpha_{i+1}}$ est attribué à celui ci. Un alignement par programmation dynamique (de type Smith et Waterman (Smith and Waterman, 1981)) est ensuite effectué sur les suites de symboles des deux structures. La structure est donc codée en une chaîne de caractères sur un alphabet à 20 symboles. Cette description des structures est - au niveau du nombre de symboles - assez proche de la description en angles α discrétilisés que nous utiliserons par la suite (voir YAKUSA 98). Par contre, la pertinence géométrique des 20 symboles par rapport aux angles canoniques dans les structures protéiques n'est pas réellement examinée par les auteurs.

2.3.5 Les coordonnées internes : les angles

Les angles utilisés pour la comparaison de structures protéiques sont les angles (ϕ, ψ) (Levine et al., 1984) ou les angles (α, τ) (Levitt, 1976; Usha and Murthy, 1986) ou des angles dérivés de ceux ci. Assez peu de méthodes de comparaisons structurales utilisent cette description angulaire des structures.

1) Les angles (ϕ, ψ)

Dès 1977, les angles (ϕ, ψ) ont servi dans une représentation des structures visant à faciliter leur comparaison (Balasubramanian, 1977). Cette comparaison consistait simplement à construire un graphique avec en abscisse les indices des résidus et en ordonnées les valeurs des angles (ϕ, ψ) , chacun représenté par un signe différent (\triangle et ∇ respectivement). En 1989, Karpen (Karpen et al., 1989)

a comparé les distributions des $RMSD_{\phi,\psi}$ comme cela avait été fait par Remington (Remington and Matthews, 1978; Remington and Matthews, 1980) avec le $RMSD_c$.

Une méthode de comparaison structurale a été décrite par M. Levine et coll. (Levine et al., 1984). C'est une méthode rapide mais qui ne fournit pas un alignement très précis. Une matrice est construite avec dans chaque case la somme des différences des ϕ et ψ : $\Delta_{i,j} = |\phi_i - \phi_j| + |\psi_i - \psi_j|$ où i est un résidu de la protéine A et j un résidu de la protéine B. Le score général est la somme des scores

de chaque diagonale de la matrice calculé avec la formule : $\chi_N^2 = \frac{\sum_{i=1}^n -n\bar{\Delta}}{n}$ où n est le nombre de cases dans la diagonale et $\bar{\Delta}$ la moyenne des Δ . Ce score donne donc une idée de la ressemblance globale des deux structures. Il a été un peu modifié pour diminuer le poids des grandes diagonales et en enlevant certains points isolés de la matrice. Un Z-score est aussi calculé, la moyenne et l'écart-type étant estimé par génération de matrices aléatoires à partir de la matrice construite. L'alignement final est déterminé en utilisant la méthode du programme HOMOLOGY (Rao and Rossmann, 1973; Rossmann and Argos, 1975), c'est-à-dire une superposition des résidus correspondants en minimisant le $RMSD_c$.

2) Les angles (α, τ)

Les angles (α, τ) ont été encore moins utilisés que les angles (ϕ, ψ). Cependant, la description utilisée par S. Rackovsky et coll. utilise des concepts qui en sont assez proches. (Nishikawa et al., 1974; Rackovsky and Scheraga, 1978; Rackovsky and Scheraga, 1980; Rackovsky and Scheraga, 1981; Rackovsky and Scheraga, 1982; Rackovsky and Goldstein, 1988). En effet, ces auteurs définissent une courbure et une torsion locale pour chaque fragment de 4 résidus chevauchants. Dans cette série d'articles, ils ont fait une large étude de la conformation des fragments structuraux. Par exemple, ils ont calculé la distribution de ces courbures discrétisées en 18 classes (Rackovsky and Scheraga, 1978; Rackovsky and Scheraga, 1980). Des protéines fonctionnellement similaires posséderaient des distributions similaires, même quand leur structure globale diffèrent (Rackovsky and Goldstein, 1988).

Roterman et coll. (Roterman, 1995; Leluk et al., 2003) considèrent les structures protéiques comme des hélices avec un rayon de courbure variable ; ils utilisent le même genre d'angles c'est-à-dire l'angle V formé par les plans de deux liaisons peptidiques et la courbure R définie sur 5 résidus. Les distributions de ces deux paramètres sont comparées. Un score de similarité entre le résidu i d'une structure et le résidu j d'une autre est calculé tel que $S_{i,j} = \frac{P_i - P_j}{\min(P_i, P_j)}$ où $P = \ln(R)$ ou $\ln(V)$. Lorsque $S < 50\%$ les pentapeptides sont considérés comme similaires. Ce score est généralement calculé sur des fenêtres de 25 amino-acides.

Usha et Murthy

Usha et Murthy (Usha and Murthy, 1986) ont eu une autre approche avec les angles (α, τ) réels. Ils calculent les angles (τ_i, α_i) sur des fragments de 7 résidus ; ils obtiennent donc 4 angles de liaison (τ) et 3 angles de torsion (α) par fragment. Les auteurs calculent la somme des différences entre ces angles

sur deux protéines connues pour être similaires (les chaînes *alpha* et *beta* de l'hémoglobine). Ils tirent de cette distribution la probabilité que la différence soit supérieure à x ($p = (\text{nombre de différences} > x) / \text{nombre total de paires}$) pour deux fragments dans une région structurale similaire. Pour comparer deux structures, ils font ensuite un alignement par programmation dynamique des deux structures (découpées en fragments). Le score de la paire de fragments i, j est le produit des probabilités associées au paires d'angles de ce fragment.

2.3.6 Méthodes utilisant plusieurs descriptions

1) COMPARER

Cette méthode (Sali and Blundell, 1990; Zhu et al., 1992) pourrait être classée dans d'autres sections car elle utilise toutes sortes d'informations à tous les niveaux. Ces niveaux sont : résidus, SSE, *super secondary structure*, domaine et protéine entière. Toutes ces informations sont prises en compte dans un score utilisé par un algorithme de programmation dynamique (de type NWS (Needleman and Wunsch, 1970)). Pour résumer, ce score est la somme pondérée de chaque score calculé individuellement pour chaque propriété ou relation, et un grand nombre de propriétés ou relations participent au score. Les propriétés comparées entre deux résidus i et j de deux structures sont : la nature des résidus, certaines propriétés physico-chimiques (Argos, 1987), l'orientation de la chaîne latérale, l'accessibilité de la chaîne principale, de la chaîne latérale, les angles (ϕ, ψ), la distance entre les résidus après superposition⁶, l'orientation de la chaîne principale après superposition des deux résidus... Les structures secondaires sont attribuées par DSSP et les propriétés associées sont pratiquement les mêmes que celles des résidus. Tous les scores - qui sont relativement simples - ne seront pas détaillé ici mais par exemple, pour l'angle ϕ , le score est $w_{i,j} = |\phi_i - \phi_j|$ (différence circulaire). D'autres scores sont aussi calculés selon les « relations », prenant en compte, par exemple, le nombre de liaisons hydrogènes auxquelles participent les deux résidus i et j .

Lorsque tous les paramètres ont été calculés, ils sont utilisés pour une étape finale d'alignement par programmation dynamique (NWS).

2) Méthodes utilisant la programmation dynamique

Le programme SSAP et ses dérivés

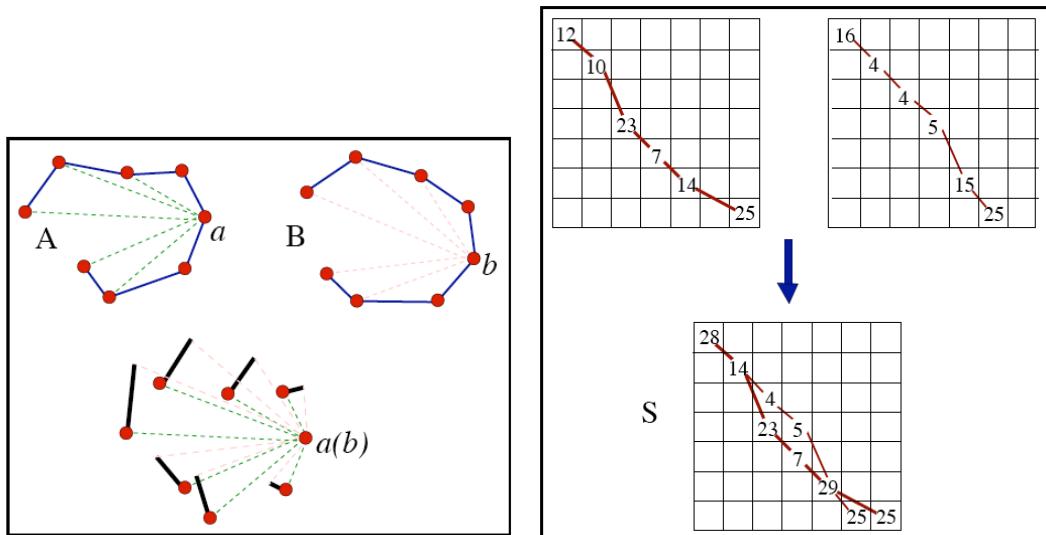
Le programme SSAP (*Structure and Sequence Alignment Program*) (Taylor and Orengo, 1989b; Taylor and Orengo, 1989a), nommé ensuite SAP (*Structure Alignment Program*) (Taylor, 1999) est utilisé pour établir la classification CATH (Orengo et al., 1997). L'algorithme de « double » programmation dynamique (DDP pour *Double Dynamic Programming* (Orengo and Taylor, 1990)) est utilisé dans toutes les versions. La première version (SSAP) utilise des informations à la fois de séquence, de géométrie de la structure du squelette peptidique (distances internes) et des propriétés physico-chimiques des résidus. La seconde version (SAP) ne prend en compte que les informations de géométrie. Une

⁶La méthode de superposition est MNYFIT (Sutcliffe et al., 1987)

version effectuant l’alignement local de structures a été développée (Orengo and Taylor, 1993) en adaptant l’algorithme de Smith et Waterman (Smith and Waterman, 1981). Une autre version a été développée pour l’alignement multiple de structures (Taylor et al., 1994). Une dernière version travaille au niveau des SSE (Orengo et al., 1992), elle est assez rapide pour permettre la recherche de similarité structurale dans une banque.

La double programmation dynamique

Pour expliquer la double programmation dynamique, la description des structures sera uniquement faite à l’aide de leurs distances internes. Deux matrices de distances internes sont définies, une pour la protéine A et une pour la protéine B. L’alignement est réalisé en deux étapes : pour chaque C_α de la protéine A et de la protéine B, un alignement par NWS est calculé en utilisant les distances internes. Tous ces alignements optimaux sont stockés dans une matrice à partir de laquelle l’alignement optimal final est calculé(figure 2.8).



- (a) Les lignes continues bleues représentent les squelettes peptidiques des deux structures A et B et les points rouges les C_α . Les lignes pointillées sont les deux axes internes prises en compte pour la paire de résidus mal trouvé dans chaque matrice en haut est accumulé de (a,b). Dans le schéma du bas, les deux structures sont dans la matrice « somme » finale en bas. L’alignement superposées et les lignes continues noires représentent final est trouvé en utilisant à nouveau la programmation dynamique avec cette matrice finale.
- (b) Illustration de l’algorithme de double programmation dynamique : dans chacune des trois matrices, les deux axes sont les deux structures. L’alignement optimales internes prises en compte pour la paire de résidus mal trouvé dans chaque matrice en haut est accumulé de (a,b). Dans le schéma du bas, les deux structures sont dans la matrice « somme » finale en bas. L’alignement superposées et les lignes continues noires représentent final est trouvé en utilisant à nouveau la programmation dynamique avec cette matrice finale.

FIG. 2.8 – Ces deux schémas sont extraits de la revue de Wang (Wang, 2005)

Ces méthodes (Taylor and Orengo, 1989b; Taylor and Orengo, 1989a; Orengo et al., 1992; Orengo and Taylor, 1996; Taylor, 1997; Taylor, 1999) sont parmi les premières à utiliser les matrices de distances internes ou de contacts. Les auteurs utilisent deux modes de représentations des structures : les distances internes et les vecteurs entre C_β (vecteur $V_{i,j}$ entre les C_β i et j d’une structure). Ils

définissent trois vecteurs pour chaque résidu : un vecteur x pour N-C, un vecteur y pour C_β -H de longueur égale à $C_\beta - C_\alpha$ et un vecteur z perpendiculaire à x et y , enfin y est rendu perpendiculaire à x et z .

Les auteurs prennent toutes les paires de résidus (i, k) des protéines A et B respectivement et calculent les distances (après réorientation de ces vecteurs selon les résidus voisins) de toutes les paires de vecteurs partant des résidus i et k ($V_{i,j}^A, V_{k,l}^B$). On obtient ainsi une matrice de distances et dans laquelle on cherche le meilleur chemin par NWS. Les valeurs des cases de la matrice pour ce meilleur chemin sont ajoutées à une matrice qui contiendra donc à la fin la somme de tous les meilleurs chemins comme expliqué ci dessus. Les auteurs utilisent la même méthode de double programmation dynamique pour les distances internes.

2.4 Description en éléments de structures secondaires

Les structures secondaires sont des structures régulières très fréquentes dans les protéines. Elles sont très souvent considérées comme les fragments structuraux les mieux conservés dans les structures similaires ou homologues. De plus, comme ce niveau de représentation engendre moins d'éléments (de l'ordre d'une dizaine de SSE pour une protéine de 300 résidus), les comparaisons de structures sont plus rapides que les précédentes. Les méthodes de comparaison travaillant à ce niveau sont donc souvent utilisées pour la recherche de similarités structurales dans une banque de structures. Les résultats seront en contrepartie moins précis que ceux des méthodes vues précédemment, et certaines régions des protéines sont ignorées. Les structures secondaires prises en compte par ces méthodes sont généralement uniquement les hélices α et les feuillets β . Il existe plusieurs méthodes d'attribution des éléments de structure secondaire. La plus courante est DSSP (Kabsch and Sander, 1983).

Il faut alors trouver un mode de représentation adapté. Le plus utilisé est la représentation de chaque SSE par un vecteur. Tous les paramètres de distances, d'angles, *etc.* qui sont calculés dans ces méthodes peuvent mettre en jeu soit deux SSE d'une même structure soit deux SSE de structures différentes. Pour éviter toute confusion, je nommerai « paire de SSE » les paires de SSE d'une même protéine et « SSE en équivalence » (« en correspondance » ou « alignés ») les paires de SSE de structures différentes.

2.4.1 Méthodes avec les SSE représentés par des vecteurs

La représentation en vecteur

La représentation la plus fréquemment usitée pour un SSE est un vecteur colinéaire à l'axe principal. L'axe principal est la droite pour laquelle l'inertie des éléments (souvent uniquement les C_α) du SSE est minimale (voir figure 2.9). Le vecteur est orienté de l'extrémité N-terminale vers l'extrémité C-terminale et les extrémités sont les projections des atomes aux extrémités du SSE.

Un ensemble de vecteurs représente une structure, le problème est donc de comparer deux ensembles de vecteurs. L'ordre des SSE dans la chaîne peptidique peut ou non être pris en compte lors de la comparaison.

Méthodes fondées sur la théorie des graphes

Les algorithmes les plus utilisés pour la comparaison des ensembles de vecteurs proviennent de la théorie des graphes.

Méthode générale de comparaison.

Les procédures se déroulent en quatre étapes :

étape 1 assignation des structures secondaires et calculs des paramètres des vecteurs ;

étape 2 construction pour chaque structure d'un graphe ayant pour sommets les SSE. Les arêtes peuvent être définies de différentes manières mais y sont toujours associées des informations

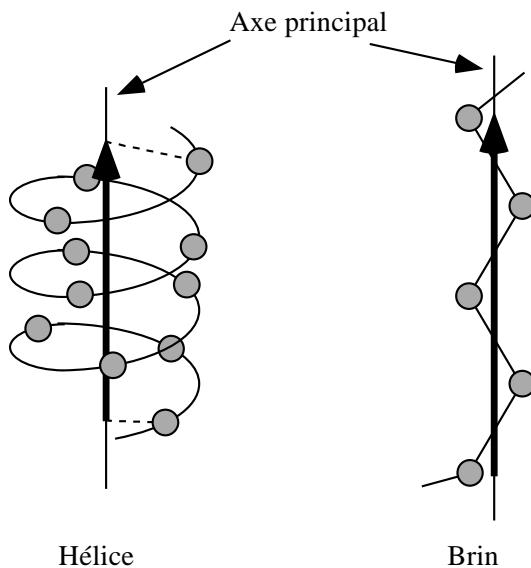


FIG. 2.9 – Axes principaux des deux éléments de structure secondaire les plus courants : hélice α et feuillet (brin) β .

sur l’organisation spatiale relative des vecteurs ;

étape 3 comparaison des graphes et recherche des sous-graphes similaires communs ;

étape 4 retour au niveau peptidique et alignement résidu par résidu (facultatif).

Comme aux niveaux de comparaison précédents, deux méthodes de recherche des sous-graphes communs sont possibles : rechercher les sous-graphes isomorphes par exemple avec la méthode d’Ullmann (Ullmann, 1976) ou construire un graphe de correspondance et rechercher les cliques maximales.

D. Rice, P. Willett et coll. ont développé plusieurs méthodes fondées sur la théorie des graphes, que ce soit pour comparer les structures au niveau atomique (Jakes and Willett, 1986; Brint and Willett, 1987; Clark et al., 1991; Poirrette et al., 1991; Artymiuk et al., 1992), peptidique (chaînes latérales sous forme de « pseudo-atomes » (Artymiuk et al., 1994)) ou des structures secondaires (Mitchell et al., 1990; Artymiuk et al., 1990; Grindley et al., 1993). Ils ont implémenté beaucoup d’algorithmes de comparaison de graphes, et Willet a écrit une revue (Raymond and Willett, 2002) des différents algorithmes qui ont été appliqués à la comparaison de structures protéiques. Le programme POS-SUM (Mitchell et al., 1990; Artymiuk et al., 1990) (Protein Online Substructure Searching) recherche des motifs structuraux définis par l’utilisateur, en utilisant des représentations sous forme de graphes des éléments de structures secondaires. Il utilise une version légèrement modifiée de l’algorithme de Kabsch et Sander (Kabsch and Sander, 1983) pour assigner les régions de structures secondaires.

L’axe principal des SSE est calculé d’après la position des résidus qui les composent, et puis les angles et les distances entre axes sont calculés. Ces données sont stockées dans une matrice, qui représente donc la structure protéique. Toutes les protéines dans la PDB sont codées en utilisant ce procédé

et une représentation semblable est employée pour le motif recherché. La méthode de comparaison des graphes est alors une version modifiée de l'algorithme de Ullmann (Ullmann, 1976) pour la recherche de sous-graphes isomorphes. La différence avec le programme PROTEP (Grindley et al., 1993) se trouve dans l'algorithme de comparaison : dans ce dernier, un graphe de correspondance est construit et les cliques maximales sont recherchées comme dans la méthode de Bron et Kerbosch (Bron and Kerbosch, 1973). PROTEP peut prendre en compte ou non l'ordre séquentiel des SSE. La méthode de PROTEP a été reprise dans le programme GRATH (Harrison et al., 2003) qui recherche les similarités entre une nouvelle structure et une banque de domaines structuraux (domaines de CATH).

Comme dans PROTEP, la méthode de comparaison du programme VAST (Gibrat et al., 1996; Madej et al., 1995) implique la construction d'un graphe de correspondance (l'ordre des SSE sur la structure est pris en compte). Une procédure de recherche exhaustive, basée sur un algorithme rapide de la théorie des graphes, identifie les sous-structures grossièrement similaires. Le $RMSD_c$ obtenu par la superposition des paires de SSE est comparé à la distribution observée lorsque qu'une telle paire est tirée au hasard dans la banque de structures connues, ce qui donne une probabilité, p_c . La significativité d'un résultat est évaluée en multipliant p_c par le nombre d'alignements alternatifs possibles de sous-structures pour la paire de structure considérée. L'alignement-superposition optimal des C_α est recherché avec une procédure d'échantillonnage de Monte-Carlo qui raffine l'alignement précis des résidus et les limites des SSE conservés.

Les graphes sont aussi utilisés par plusieurs auteurs (Rufino and Blundell, 1994; Mizuguchi and Go, 1995). Dans la méthode récente SSM (Krissinel and Henrick, 2003; Krissinel and Henrick, 2004a; Krissinel and Henrick, 2004b), les structures secondaires sont assignées par PROMOTIF (Hutchinson and Thornton, 1996) et les paramètres associés aux paires de vecteurs sont - comme pour les méthodes précédentes - les angles et les distances entre deux vecteurs (d'une même structure). La méthode pour comparer les graphes est inspirée de l'algorithme de Ullmann. L'ordre des structures secondaires peut être différent dans l'alignement final des deux structures.

2.4.2 Autres méthodes de comparaison

1) Méthodes de classifications

La méthode de comparaison de SARF (Alexandrov et al., 1992; Swindells and Alexandrov, 1994; Alexandrov and Go, 1994; Alexandrov and Fischer, 1996; Alexandrov, 1996) ressemble plus à celle du programme CE (voir section « Les coordonnées internes »), sauf qu'elle travaille au niveau des SSE (ceux-ci étant toujours représentés par des vecteurs). Les structures secondaires sont assignées uniquement par rapport aux C_α en comparant les fragments locaux avec certaines des structures canoniques définies par Unger et coll. (Unger et al., 1989). Les vecteurs sont calculés comme dans le programme Molscript (Kraulis, 1991). Pour chaque paire de SSE dans une même structure, 5 paramètres sont calculés : un angle entre les deux vecteurs et quatre distances (distances des extrémités des SSE à la bissectrice de l'angle, voir figure 2.10).

La comparaison de deux protéines consiste à rechercher les ensembles de SSE similaires entre

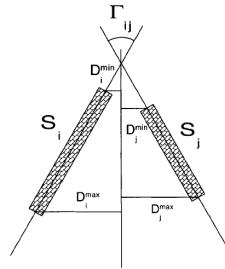


FIG. 2.10 – Les 5 paramètres associés aux vecteurs représentant les SSE dans le programme SARF. La figure est extraite de l'article d'Alexandrov et Fischer(Alexandrov and Fischer, 1996). Γ_{ij} est l'angle entre les deux vecteurs S_i et S_j . Les quatre autres paramètres ($D_i^{\min}, D_i^{\max}, D_j^{\min}, D_j^{\max}$) sont les distances minimales et maximales entre les extrémités des vecteurs et la bissectrice de l'angle Γ_{ij}

les deux structures. Ces ensembles sont construits en ajoutant deux SSE en correspondance (un sur chaque structure) si ceux-ci ont des paramètres similaires avec tous les SSE déjà dans l'ensemble. Pour deux SSE en correspondance à ajouter l'ensemble, il faut donc par exemple que tous les angles formés par le nouveau SSE et tous ceux déjà présents de la protéine A soient proches de tous les angles formés par le nouveau SSE et les SSE de l'ensemble dans la protéine B. Pour limiter l'espace de recherche, il faut que les nouveaux SSE soient à moins de 25 Å de tous ceux de l'ensemble. Ensuite a lieu une étape de raffinement de l'alignement au niveau des C_α .

La superposition initiale est effectuée avec les vecteurs en correspondance. L'alignement des C_α est ensuite trouvé par une méthode itérative superposition-alignement, l'alignement étant déterminé par programmation dynamique. Le score final permettant d'évaluer la significativité est $S = \frac{4N}{2 + RMSD_c}$ où N est la longueur de l'alignement. Comme SARF est utilisé pour comparer une structure et toutes celles d'une banque, il est possible de calculer un Z-score pour le score S avec la moyenne et l'écart type des scores de la protéine requête et des protéines de la banque. Cependant, lorsque deux protéines A et B sont comparées, les distributions des scores ne sont pas les mêmes pour les protéines A et B et le Z-score n'est donc généralement pas le même. Le score de similarité des protéines A et B est donc plutôt la somme des deux Z-scores.

La méthode PSI (Camoglu et al., 2003) a aussi pour objectif de rechercher les structures similaires dans une banque. Un index des structures de la banque est construit en 4 étapes :

1. assignation des SSE et approximation des SSE en segments ;
2. construction de triplets : pour chaque SSE, les 4 SSE les plus proches (parmi les SSE à moins de 50 Å) sont choisis et vont donc former 6 triplets avec le SSE de départ. Les triplets sont donc composés de 3 SSE d'une même structure ;
3. extraction des caractéristiques des vecteurs : pour chaque triplet, 15 points à égale distance sont définis sur le segment et 3 valeurs sont calculées pour chaque paire de SSE : la distance minimale entre 2 points, la distance maximale et l'angle entre 2 segments. Il y a donc 9 valeurs pour chaque triplet ;
4. indexation multidimensionnelle des structures : chaque triplet est stocké dans un objet qui

contient aussi la position dans la protéine de chaque SSE, le nombre de résidus de chaque SSE et le code PDB de la protéine. Ces objets sont rangés dans un arbre permettant d'y accéder rapidement.

Les paires de triplets similaires dans deux structures sont donc trouvés rapidement. Ils sont ensuite combinés pour former un alignement ou une correspondance en construisant des graphes et en cherchant les composantes connexes.

Finalement, une *p-value* est calculée pour un alignement de SSE donné. Les SSE sont représentés par des points (leurs centres de gravité) dans l'espace cartésien. Tous les alignements possibles sont générés, les points représentant les SSE alignés sont superposés et le $RMSD_c$ correspondant calculé à chaque fois. Des sphères sont alors définies dans l'espace, centrées sur chaque SSE (point) de la structure requête, et ayant pour rayon le $RMSD_c$ calculé. La probabilité associée à un alignement donné est alors le ratio entre le volume de ces sphères et le volume total de la protéine requête. La *p-value* est la somme des probabilités des alignements contenant autant de SSE que l'alignement étudié.

Le programme DEJAVU (Kleywegt and Jones, 1997) détecte les similarités structurales entre une structure requête et les structures de la banque. La similarité structurale dépend du nombre de SSE, de leurs longueurs, et des distances, angles et connectivité entre SSE. Les meilleurs ensembles de SSE similaires entre la requête et une structure de la banque sont recherchés : i) en ne comparant que les structures ayant au moins autant de SSE du même type que la requête ; ii) en sélectionnant tous les SSE qui sont similaires en termes de longueur et de nombre de résidus ; iii) en cherchant le meilleur ensemble de SSE pour chaque paire structures requête / structure de la banque par recherche exhaustive. Une étape finale superposition-alignement (par programmation dynamique) faite par LSQMAN (Kleywegt and Jones, 1995; Kleywegt, 1996) permet de trouver l'alignement au niveau des C_α . Un score - celui défini par Levitt et Gerstein (Levitt and Gerstein, 1998) - est attribué aux structures de la banque selon leur similarité avec la structure requête. Ce score est défini comme $S_{str} = M(\sum(1/(1+(d_{ij}/d_0)^2) - N_{gap}/2)$, où d_{ij} distance entre les C_α i et j en correspondance, $d_0 = 5\text{\AA}$, $M = 20$, N_{gap} le nombre de gaps dans l'alignement (gaps non comptés aux extrémités). Levitt montre que - contrairement au $RMSD_c$ - ce score suit une distribution des valeurs extrêmes, ce qui permet un calcul de *p-value* pour les alignements.

Le programme TOP (Lu, 2000) ressemble plus à une méthode itérative de superposition-alignement. Les éléments de structures secondaires sont assignés par DSSP et deux points situés aux extrémités N et C terminales sont calculés. Les paires de SSE similaires entre les deux protéines (deux SSE par protéine) sont recherchées en superposant les quatre points définis. Deux paires de SSE sont retenues si les angles entre SSE sont similaires et si la distance minimale entre les SSE est faible. Suit alors une étape de superposition-alignement où sont itérativement ajoutés les SSE proches. L'alignement au niveau des C_α est ensuite effectué grâce aussi à une méthode itérative de superposition-alignement par ajout des C_α proches (et respectant certaines contraintes par exemple sur la direction de la liaison $C_\alpha - C_\beta$). Le score de similarité final est $RMSD_c/(N_{match}/N_0)$ où N_{match} est le nombre de résidus en

correspondance et N_0 la moyenne des longueurs des protéines, ou la plus petite longueur, ou encore la longueur de la structure requête dans le cas d'une comparaison avec une banque.

2) Programmation dynamique

La programmation dynamique a été adaptée à la comparaison de structures selon leurs SSE dans LOCK (Singh and Brutlag, 1997; Shapiro and Brutlag, 2004a). Les structures secondaires sont assignées par DSSP et 7 paramètres sont calculés : 5 sont calculés entre les paires de SSE d'une même structure (angles entre deux vecteurs d'une même protéine, distances internes...) et 2 sont calculés entre deux vecteurs de protéines différentes (distance et angle entre les vecteurs des deux protéines) ; une superposition préalable est donc nécessaire. La première étape de l'alignement est de trouver les paires de paires de SSE qui sont compatibles *i.e.* dont la somme des 5 scores indépendants de l'orientation est supérieure à un seuil et dont au moins deux SSE sont contigus sur une des deux protéines (sans gap). A partir de ces quatre SSE, les deux structures sont superposées et les deux paramètres manquants sont calculés pour toutes les paires de SSE (y compris les paires de SSE de structures différentes). La programmation dynamique est alors utilisée pour trouver le meilleur alignement, le score dépendant des deux paramètres calculés après superposition. Pour le meilleur alignement, le $RMSD_c$ est minimisé et l'alignement est affiné selon la méthode classique de superposition-alignement. Le programme FOLDMINER (Shapiro and Brutlag, 2004a; Shapiro and Brutlag, 2004b) permet de construire des « profils » à partir de ces alignements de paires de structures. Ces « profils » sont décrits de manière probabiliste par rapport à la conservation des éléments de structure secondaire dans plusieurs structures trouvées comme similaires dans la banque.

La méthode de comparaison structurale de PrISM (Yang and Honig, 1999; Yang and Honig, 2000a) utilise la même méthode que SSAP (Taylor and Orengo, 1989b) : la double programmation dynamique, mais au niveau des SSE. Pour chaque paire de SSE d'indice i dans la protéine A et d'indice j dans la protéine B, la première phase d'alignement est calculé selon le score :

$$S(ij)_{mn} = \left(\frac{p}{q + |d_{im} - d_{jn}|} \right) \left(\frac{d_{im} + d_{jn}}{r} \right)^{-t} \times f(|L_{in} - L_{jn}|) \quad (2.13)$$

avec m , index d'un SSE de la protéine A, n , index d'un SSE de la protéine B, L_{in} , l'angle défini par la direction des vecteurs associés aux SSE i et n , de même pour L_{jm} , p , q , r et t sont des constantes dont les valeurs ont été optimisées empiriquement (respectivement 50Å, 2Å, 8Å et 1,7), $f(|L_{in} - L_{jn}|)$ est une fonction discrète renvoyant 2 si la différence angulaire est inférieure à 40°, 1 si elle est inférieure à 80° et 0 sinon, d_{im} est une distance interne définie telle que :

$$d_{im} = \frac{\sum_{k=1}^{l_i} \min(a_{k1}, \dots, a_{kl_m})}{l_i} \quad (2.14)$$

où l_i et l_m sont les nombres de résidus dans les SSE i et m respectivement, a_{kl} la distance entre le

$C_\alpha k$ du SSE i et le $C_\alpha l$ du SSE m .

Une fois ces alignements calculés et cumulés dans une matrice, l’alignement final est calculé par une nouvelle étape de programmation dynamique comme dans SSAP. L’alignement est ensuite affiné au niveau des résidus par une méthode itérative superposition-alignement par programmation dynamique. Finalement, un score, nommé score PSD (*Protein Structural Distance*) est calculé :

$$PSD(A, B) = \left(-\frac{\log \left(\frac{a}{\max(a,b)} \frac{s(A,B)}{s(A,A)} \right)}{\log(x)} \right)^2 + \left(\frac{RMSD_c}{y} \right)^2 \quad (2.15)$$

où a est le nombre de structures secondaires dans la structure A, b , le nombre de structures secondaires dans B, x et y des constantes. Ce score tient compte à la fois des SSE alignés et du $RMSD_c$, il permet, d’après les auteurs, d’identifier des similarités lointaines.

2.4.3 Méthodes avec SSE représentés par d’autres caractéristiques

1) Programmation dynamique sur scores de similarité basés sur un modèle d’évolution

Dans la méthode MATRAS (Kawabata, 2003), trois scores de similarités structurales sont calculés d’une manière analogue aux scores de similarité des acides aminés des matrices PAM (Dayhoff et al., 1983). Les calculs sont effectués entre paires de protéines homologues d’une banque non-redondante à 95% d’identité extraite de SCOP (Murzin et al., 1995), regroupées et alignées grâce à leurs séquences. Le score $S(i, j)$ est égal à $\log(P(j \rightarrow i)/P(i))$ estimé sur l’échantillon des protéines homologues. Comme pour les matrices PAM pour les séquences, le modèle d’évolution entre structures est ici une accumulation de transitions de Markov. Ces trois scores de similarité structurale sont :

- un score S_{SSE} , de paires de SSE, où les trois états sont les trois combinaisons d’hélices α et de feuillets β . Les SSE sont représentés comme des vecteurs. Quatre fonctions de similarité géométrique sont estimées pour chaque type de paire (θ_1 , θ_2 , d et ϕ , voir figure 2.11) ainsi que deux fonctions de similarité de longueur (L_1 et L_2). Le score S_{SSE} est la somme de ces six fonctions estimées sur l’échantillon des protéines homologues ;
- un score environnemental S_{env} où les états sont les deux structures secondaires (hélices α et feuillets β) ainsi que 3 types de boucles définis selon leurs angles (ϕ, ψ). Chacune des catégories peut être enfouie ou exposée, ce qui donne 10 états ;
- un score de distances S_{dis} où les états sont les distances internes entre C_β discrétisées avec un pas de 1Å. Les bornes sont [0, 50Å], il y a donc 50 états. Le score S_{dis} entre une distance D_{i_x} - entre le i^e et le x^e résidu d’une protéine - et la distance D_{j_y} - entre le i^e et le x^e résidu d’une autre protéine - est évalué par les transitions observées entre ces distances pour tous les intervalles k en résidus ($k = |i-x| = |j-y|$, l’intervalle est identique sur les 2 structures).

L’alignement se déroule en trois étapes :

1. premier alignement grossier des SSE en utilisant les scores S_{SSE} ; les meilleures suites de SSE sont trouvées par *branch and bound* ;

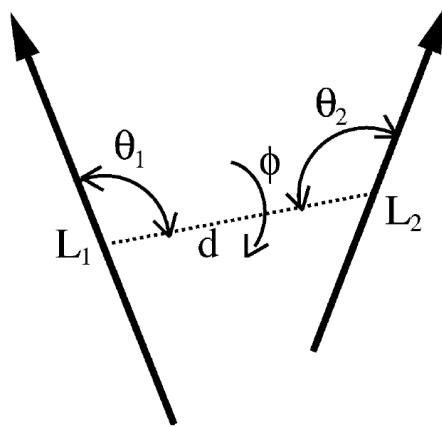


FIG. 2.11 – Les 4 paramètres géométriques associées aux paires de vecteurs de SSE de MATRAS. Figure extraite de (Kawabata, 2003). Ces paramètres sont : la distance minimale entre les deux SSE d , les deux angles (plans) θ_1 et θ_2 et l’angle dièdre ϕ entre les plans formés par chacun des deux vecteurs et la droite minimale en pointillée. Les deux longueurs sont L_1 , L_2 , exprimées en résidus.

2. un second alignement des SSE est obtenu par programmation dynamique (alignement local avec pénalités d’ouverture et d’extension de gap) en utilisant le score environnemental S_{env} et en affectant un bonus pour l’alignement des SSE déjà en correspondance à l’étape précédente ;
3. alignement des résidus : l’alignement précédent étant utilisé comme initialisation, plusieurs alignements successifs par programmation dynamique sont effectués en utilisant le score de distances S_{dis} . Le score de similarité est $S(i, j) = \frac{1}{2} \sum_a S_{dis}^k(d_{i,x(a)}, d_{j,y(a)})$ où $d_{i,x(a)}$ et $d_{j,y(a)}$ sont les distances internes entre les résidus i et $x(a)$ de la première protéine et entre les résidus j et $y(a)$ de la seconde protéine. Les résidus $x(a)$ et $y(a)$ sont fixés par l’alignement précédent. Cette étape est répétée jusqu’à convergence.

Un Z-score est calculé avec le score final de l’alignement, la moyenne et l’écart type étant calculé en comparant la structure requête avec toute la banque.

2) Algorithme génétique K2 (anciennement Kenobi)

La méthode K2 (Szustakowski and Weng, 2000; Szustakowski and Weng, 2002) travaille à la fois au niveau des SSE et des résidus mais seuls les résidus des SSE sont pris en compte dans la majorité de la procédure. En effet, la procédure comporte trois étapes : un alignement des SSE représentés comme des vecteurs avec construction d’un graphe, un alignement des résidus des SSE par algorithme génétique et une étape finale de raffinement par une méthode itérative de superposition-alignement (ajout des C_α proches d’un alignement à l’autre).

L’alignement initial des SSE est effectué avec une méthode similaire aux précédentes : les SSE sont représentés par des vecteurs. Le meilleur alignement de SSE est défini par programmation dynamique (NWS). Si l’ordre dans la séquence n’est pas obligatoire, un graphe bipartite est construit. Un graphe bipartite est un graphe qui contient deux types de sommets et dont les arêtes joignent toujours

deux sommets de types différents. Les sommets sont les SSE et ils sont de types différents suivant qu'ils appartiennent à la protéine A ou B. Les arêtes sont pondérées selon la similarité des SSE. La meilleure équivalence de SSE est alors le sous-graphe où la somme des poids des arêtes est maximale. La correspondance trouvée est vérifiée en superposant les vecteurs. Si certaines paires de SSE ne respectent pas certaines contraintes sur les angles et les distances, elles sont éliminées.

L'algorithme génétique vise ensuite à trouver l'alignement au niveau des résidus en se basant sur l'alignement des vecteurs. Un individu est une suite de paires de SSE appartenant à chacune des structures. Les SSE d'une même paire sont obligatoirement de même nature (2 hélices ou 2 feuillets). La population initiale est construite à partir des 50 meilleures correspondances de l'étape précédente. Le score des paires de SSE est calculé avec la fonction « élastique » de DALI (*cf.* équations 2.9 et 2.10). La probabilité de tirer une paire de SSE est $P = \frac{e^{S_k}}{\sum_k e^{S_k}}$ avec S_k le score « élastique » calculé avec la formule de DALI pour le SSE k . Il y a ensuite 4 types d'opérateurs :

mutate qui permet soit d'ajouter ou de retirer certains résidus, soit de décaler une des structures de 1 résidu.

hop qui échange par exemple deux hélices dans un individu ;

```

HA1    HA2    HA3
HB1    HB2    HB3
devient
HA3    HA2    HA1
HB1    HB2    HB3

```

swap deux individus peuvent échanger tout leur alignement de paires de SSE mais uniquement pour les SSE de même type (ex : échanger tout l'alignement des hélices) ;

crossover comme ci-dessus mais pas limité par le type de SSE : ils échangent un morceau de l'alignement des SSE.

L'alignement ayant le meilleur score est affiné d'abord au niveau des SSE puis il est étendu aux résidus ne faisant pas partie des SSE.

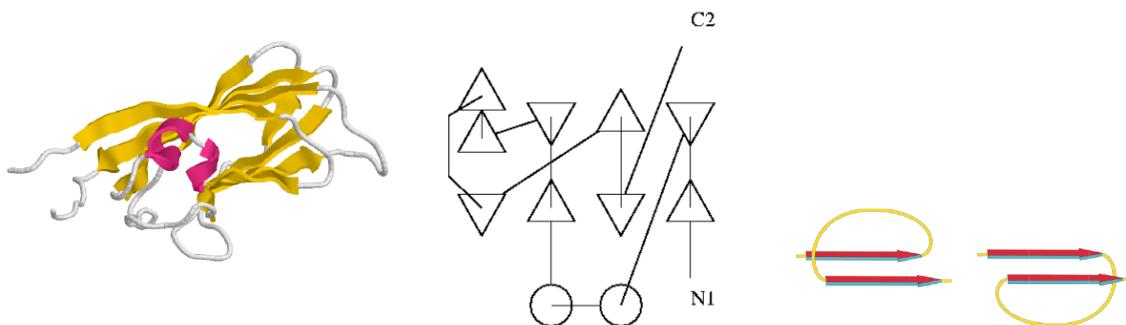
Après affinement de l'alignement des résidus par la méthode itérative, un score final est calculé toujours selon l'équation du score élastique de DALI. D'après les auteurs, la distribution de ce score suit une distribution des valeurs extrêmes dont ils ont estimé les paramètres. Une *p*-value est alors calculée.

3) Chaînes de caractères, recherche de motifs

Les méthodes suivantes sont fondées sur des algorithmes sur les chaînes de caractères.

TOPS (Gilbert et al., 1999; Gilbert et al., 2001) compare des « diagrammes topologiques » (voir figure 2.12(a)) de structures protéiques. Ces diagrammes existent depuis longtemps (Sternberg and Thornton, 1977), mais n'ont été formalisés et générés automatiquement qu'assez récemment (Flores

et al., 1994; Westhead et al., 1999). Les diagrammes de TOPS sont des représentations simplifiées des structures comme une suite de SSE prenant en compte la connectivité des SSE, leurs directions et les contacts entre SSE. Ils contiennent également des informations sur les liaisons hydrogène et la chiralité entre brins *beta* parallèles appartenant au même feuillet ou tonneau *beta* (non explicitement visibles sur les schémas, mais voir figure 2.12(b)). Chaque type de SSE est représenté par un symbole (α et β) mais la direction générale, vers le haut (+) ou vers le bas -, est aussi stockée dans le symbole. L'alphabet est donc $\Sigma = \alpha_+, \alpha_-, \beta_+, \beta_-$. Deux relations (symétriques) sont ensuite possibles entre symboles : la relation liaisons hydrogène (lorsque plusieurs liaisons hydrogène sont présentes entre deux SSE) et la chiralité.



(a) Structure ISTM visualisée avec Rasmol et sous la forme de diagramme TOPS. Cette structure est composée de motifs en clé grecque qui adoptent une organisation en sandwich de feuillets β . Les hélices α sont représentées par des ronds et les brins β par des triangles. Figure extraite de *Annual Report 2000, University of Leeds* accessible à l'adresse www.astbury.leeds.ac.uk/Report/2001/Report/62westhead.prot.topology.pdf.

(b) A droite, une connexion droite entre 2 brins β (configuration très fréquente), à gauche, une connexion gauche (configuration rare).

FIG. 2.12 – (a) Diagramme de TOPS et (b) connexion droite ou gauche.

Pour comparer un diagramme à ceux d'une banque, ces derniers sont assouplis en permettant des gaps de SSE. Une structure requête est donc convertie en diagramme puis est comparée aux diagrammes assouplis de la banque avec un algorithme inspiré des algorithmes de recherche de sous-graphes isomorphes tirant parti de l'ordre des SSE.

TOPSCAN

Dans TOPSCAN (Martin, 2000), les structures protéiques sont converties en suites de symboles. Chaque symbole représente un élément de structure secondaire assorti d'autres paramètres comme la direction, la longueur, l'accessibilité, la proximité des SSE (tous ces paramètres sont discrétisés). La technique d'alignement utilisée sur ces « chaînes de caractères topologiques » est la programmation dynamique (NWS). La similarité entre les symboles topologiques utilisée comme score dans NWS est relativement arbitraire (non fixée sur un échantillonnage). TOPSCAN est une méthode rapide, mais approximative. Elle est en fait utilisée comme un filtre avant un alignement par SSAP qui est trop lent pour permettre une recherche sur banque.

2.4.4 Alphabets structuraux

Les structures secondaires prises en compte ici (hélices α et feuillets β) ne représentent cependant qu'environ 50% des structures, le reste étant nommé « boucles » et constituant un groupe non homogène de structures non régulières. Plusieurs études visent à caractériser plus finement les petits blocs structuraux (moins de 20 résidus). La plupart de ces travaux ont pour objectif de reconstruire la structure d'une protéine uniquement avec l'information de séquence. J'aborderai ici uniquement les méthodes utilisées aussi pour la comparaison de structures protéiques. Pour un résumé complet de ces méthodes voir (de Brevern et al., 2001).

Les blocs structuraux sont le plus souvent de taille fixe, comportent quelques résidus (moins de 10) et sont chevauchants. Ils sont classés en comparant toutes les sous-structures d'une taille donnée d'une banque non redondante de structures. Ces classes de blocs partitionnent le mieux possible l'espace des structures. Un symbole est très souvent affecté à chaque classe de blocs, c'est pourquoi on parle d'alphabets structuraux.

Pour définir les blocs structuraux, Matsuo et Kanehisa (Matsuo and Kanehisa, 1993) ont découpé 93 structures de la PDB en blocs de 7 résidus chevauchants. Ils les ont ensuite regroupés selon le $RMSD_c$ de la manière suivante : un fragment est tiré au sort, il constitue le premier groupe ; ensuite, pour chaque fragment, si son $RMSD_c$ avec tous les fragments d'une classe est inférieur à un seuil, il est ajouté à ce groupe, s'il ne respecte cette condition pour aucun groupe, il constitue un nouveau groupe. Un fragment représentatif est choisi pour chaque groupe (le premier constituant le groupe). Un symbole est assigné à chaque groupe.

Les blocs structuraux étant définis, il suffit ensuite de comparer tous les fragments d'une structure donnée aux fragments représentatifs pour convertir la structure en symboles. Deux structures converties en symboles sont ensuite alignées par programmation dynamique.

Une méthode plus récente et plus étayée statistiquement a été présentée par Camproux et coll. (Camproux et al., 1999). Elle est nommée SA-Search et est disponible sur plateforme « Ressource Parisienne en Bioinformatique Structurale » (RPBS) (Guyon et al., 2004). Un modèle de loi normale multivariée a permis de discréteriser le squelette peptidique en classes de blocs structuraux de longueur 4 résidus (un critère d'information est utilisé pour déterminer le nombre de classes optimales). Les blocs structuraux - symboles d'un alphabet structural - sont décrits par 4 distances internes (voir figure 2.13). Cet alphabet structural comporte 27 symboles et permet une description précise de l'ensemble des structures (Camproux et al., 2004).

Après cette étape, l'enchaînement des blocs structuraux est évalué par un modèle de Markov entraîné sur une banque non redondante.

Pour comparer deux structures, celles-ci sont d'abord converties en suites de symboles de l'alphabet structural avec l'algorithme de Viterbi (Baum et al., 1970) qui permet de déterminer la meilleure série de symboles en prenant en compte la dépendance Markovienne. La comparaison des deux structures est ensuite faite soit par programmation dynamique (Smith et Waterman), soit par construction d'un arbre de suffixes pour rechercher des motifs exacts. Ces deux méthodes sont assez rapides pour

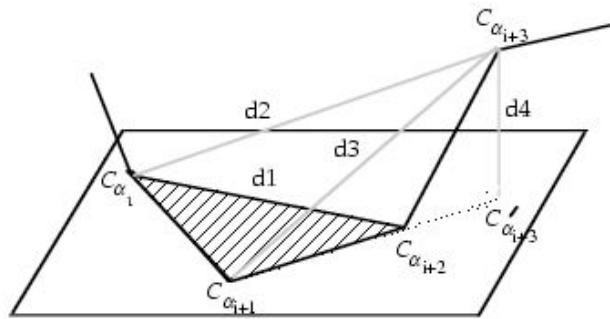


FIG. 2.13 – Chaque fragment de 4 résidus est représenté par un vecteur de taille 4 qui contient les trois distances entre C_α non consécutifs ($d_1 = d(C_{\alpha_i} - C_{\alpha_{i+2}})$, $d_2 = d(C_{\alpha_i} - C_{\alpha_{i+3}})$, $d_3 = d(C_{\alpha_{i+1}} - C_{\alpha_{i+3}})$) et la projection orthogonale d_4 de $C_{\alpha_{i+3}}$ sur le plan formé par les trois premiers C_α . Figure extraite du site <http://bioserv.rpbs.jussieu.fr/Help/SAHelp.html>

rechercher les similarités structurales dans une banque.

CHAPITRE 3

Comparaison multiple de structures

Les méthodes de comparaison multiple de structures s'appuie souvent sur des méthodes de recherche de motifs. En effet, leur objectif est de déterminer les éléments communs à plusieurs structures, ces éléments formant alors le(s) motif(s) caractéristique(s) du groupe de structures. De telles méthodes ont beaucoup été utilisées au niveaux des séquences, mais il est plus difficile de les appliquer au niveau des structures. Initialement, les méthodes d'alignement multiple de structures ont été développées pour comparer les modèles obtenus par RMN, afin de trouver le cœur structural rigide d'une structure. Cependant, lorsque des structures différentes sont comparées, comme pour les comparaisons de paires de structures, la définition de correspondances entre éléments est nécessaire.

Un objectif des méthodes de comparaison de plusieurs structures est de déterminer des régions ou blocs structuraux conservés dans une familles de structures homologues. De tels blocs structuraux sont utilisés notamment pour la modélisation par homologie ou par reconnaissance de repliements. Plusieurs serveurs sont consacrés aux alignements structuraux multiples, ils seront détaillés dans le chapitre « Les classifications existantes ». La caractérisation de ces blocs structuraux conservés peut apporter aussi des informations sur les mécanismes d'évolution des structures protéiques ainsi que sur le repliement.

Comme la plupart des méthodes de comparaison multiple (Eidhammer et al., 2000) travaillent au niveau des résidus, il me semble plus intéressant de classer les méthodes selon les méthodes de comparaison. Celles-ci peuvent être regroupées en deux grandes catégories : les méthodes procédant à des comparaisons de paires de structures et les méthodes réellement multiples procédant sans comparaison des paires de structures. De plus, il existe des méthodes mixtes construisant des motifs ou des profils à la fois au niveau de la séquence et de la structure que j'aborde pour commencer.

3.1 Enrichissement structural de motifs de séquences

Plusieurs études ont enrichi au niveau structural les motifs de PROSITE (Bucher and Bairoch, 1994). Les motifs PROSITE, définis au niveau des séquences d'acides aminés, ont été recherchés dans les séquences des structures résolues. Chaque motif correspond généralement à des sous-structures similaires (Kasuya and Thornton, 1999; Jonassen et al., 2000). Les motifs de PROCAT⁷ (Wallace et al., 1996; Wallace et al., 1997) sont définis manuellement pour des sites actifs d'enzymes : ces motifs contiennent les positions de certains atomes des résidus du motifs, ainsi que les types possibles de résidus. D'autres informations peuvent être ajoutées au motif comme l'ordre séquentiel des résidus (Jonassen et al., 1999) avec une taille maximale de gap permise (Fetrow and Skolnick, 1998)⁸. Il est donc difficile de définir ce qu'est un motif structural, car celui-ci peut contenir des informations très différentes. Des profils, contenant à la fois de l'information de séquence et de structure - notamment l'information sur les résidus conservés dans les surfaces -, ont aussi été construits (Gribskov et al., 1987; de Rinaldis et al., 1998).

⁷Site de PROCAT : <http://www.biochem.ucl.ac.uk/bsm/PROCAT/PROCAT.html>

⁸Cette liste n'est pas exhaustive.

Certaines études ne se sont pas attachées à la définition de motifs, mais à la recherche des résidus invariants dans des structures similaires (Yang and Honig, 2000b). Ces résidus peuvent faire partie d'un site actif ou d'un site de liaison et être donc indispensables à la fonction de la protéine. D'autres résidus peuvent être invariants - comme par exemple les cystéines qui permettent de former les ponts disulfures - car nécessaires au bon déroulement du processus de repliement. Les résultats sont extrêmement variables selon le jeu de données et il semble, par exemple pour les immunoglobulines, très difficile de trouver des acides aminés invariants (Yang and Honig, 2000b), hormis ceux des sites actifs.

3.2 Méthodes avec comparaison des paires de structures

Plusieurs méthodes de comparaison de paires de structures abordées au chapitre précédent ont été adaptées à la comparaison structurale multiple. Elles peuvent être subdivisées en trois sous-catégories : les méthodes avec une structure pivot (moyenne ou réelle), les méthodes d'alignement hiérarchique (avec construction d'un dendrogramme) et les méthodes utilisant les graphes. Des méthodes de superposition multiple ont aussi été développées mais elles nécessitent une correspondance ou un alignement préalable.

Il faut noter que dans un alignement optimal multiple, toutes les paires de structures ne sont pas forcément alignées de manière optimale. Ces méthodes - qui s'appuient sur les alignements optimaux par paires - sont donc des heuristiques.

3.2.1 Méthodes de superposition multiple avec correspondance donnée

Comme pour la comparaison de deux structures protéiques, les premières méthodes de comparaison de plusieurs structures superposent de manière optimale plusieurs structures où l'équivalence des résidus est fixée.

Les premières méthodes de superposition multiple ont été conçues pour mesurer la similarité et la dispersion des modèles structuraux fournis par la RMN (Shapiro et al., 1992). Les modèles étaient alors comparés soit à une structure moyenne soit à une structure référence par des méthodes de superposition optimale de deux molécules (Zuker and Somorjai, 1989; Kabsch, 1976; Kabsch, 1978; Sippl and Stegbuchner, 1991; McLachlan, 1982; McLachlan, 1979; Lesk, 1986). Cependant, aucune de ces solutions n'étant satisfaisante, des méthodes de superposition multiple ont été développées.

La solution proposée par Gerbert et Muller (Gerber and Muller, 1987) pour superposer M objets rigides constitués de N points en correspondance se déroule en deux étapes. Une structure de référence - la moins « linéaire » d'après son moment d'inertie - est d'abord choisie, et toutes les structures sont superposées avec cette structure pivot. La seconde étape consiste à minimiser :

$$E = \frac{1}{2} \sum_{n < m}^M v^{nm} \sum_{l=1}^N w_l (O^m x_l^m - O^n x_l^n)^2 \quad (3.1)$$

où M sont les molécules dont les N atomes en correspondance ont pour coordonnées x_l^m (ou x_l^n), où $m = 1, \dots, M$, $n = 1, \dots, M$, et $l = 1, \dots, N$; O^m (ou O^n) est la transformation rigide - matrice orthogonale - appliquée à la molécule m (ou n); v^{nm} est la pondération spécifique à la paire de structure n, m (en général 1), et w_l la pondération appliquée à la paire d'atomes l (en général 1 aussi). Cette seconde étape est donc la minimisation de la somme de toutes les paires de $RMSD_c$ à la fois (en gardant la M^e molécule fixe dans l'espace). La première étape permet en général de trouver rapidement une solution proche éloignée de la solution optimale qui sert de point de départ à la seconde étape (si les structures sont toutes assez proches et si la structure pivot de départ est bien choisie la plus non linéaire possible).

Pour Shapiro et coll. (Shapiro et al., 1992), l'objectif est de minimiser la même fonction que Gerber (Gerber and Muller, 1987) en utilisant l'algorithme de Berge (Berge, 1977; Shapiro and Botha, 1988). L'idée est de minimiser E itérativement en variant chacune des matrices orthogonales l'une après l'autre tout en gardant les autres matrices constantes. Evidemment, les auteurs soulignent qu'il peut exister de multiples minima locaux.

Une autre méthode, COMSUP, présentée par Kearsley (Kearsley, 1990) nécessite une structure de référence et utilise les quaternions. Dans cette méthode itérative, les M structures sont réorientées simultanément. Alternativement, Diamond, (Diamond, 1992) propose aussi une procédure de superposition multiple basée sur les quaternions mais où, à chaque itération, une structure est orientée par rapport à toutes les autres. Pour le problème des multiples minima, l'approche adoptée consiste à rechercher des minima alternatifs en variant l'ordre des rotations trouvées pour modifier le plus possible les orientations tout en modifiant le moins possible E . De plus, Diamond a utilisé cette méthode dans une procédure de classement de structures RMN avec correspondance fixée (Diamond, 1995).

Sutcliffe et coll. (Sutcliffe et al., 1987) ont développé une méthode, MNYFIT, où une structure moyenne est construite à partir des superpositions des paires de structures.

3.2.2 Méthodes d'alignement hiérarchique et méthodes de structures moyennes

Pour toutes ces méthodes (Sali and Blundell, 1990; Russell and Barton, 1992; Ding et al., 1994; May and Johnson, 1995; Taylor et al., 1994; Kawabata, 2003; Ye and Godzik, 2005; Lupyan et al., 2005), les structures sont représentées au niveau des résidus et la procédure générale est la suivante :

- étape 1 : alignement de toutes les paires de structures ($N(N - 1)/2$ alignements) ;
- étape 2 : construction d'un dendrogramme à partir des scores calculés ;
- étape 3 : alignement de toutes les structures dans l'ordre indiqué par le dendrogramme (en commençant par la paire de structures les plus similaires).

Le programme COMPARER (Sali and Blundell, 1990) représente les résidus par de multiples paramètres (voir « Comparaison de deux structures » page 51). Dans la première étape, toutes les $N(N - 1)/2$ paires de structures sont alignées avec COMPARER et un dendrogramme est construit avec les distances issues des scores d'alignement avec le programme KITSCH de PHYLIP (Felsenstein, 1985). Dans la troisième étape, l'alignement d'une structure avec un ensemble de structures déjà

x correspond à la moyenne des scores de x avec chacun des symboles d'une colonne de l'ensemble des structures déjà alignées.

Chronologiquement, la seconde méthode d'alignement de ce type est STAMP (Russell and Barton, 1992). Cette méthode est assez proche de la précédente. Les structures sont représentées uniquement par les coordonnées cartésiennes des C_α . La méthode d'alignement est celle de Barton and Sternberg (Barton and Sternberg, 1987) et les blocs sans gap de résidus sont utilisés pour superposer les paires de structures. Le $RMSD_c$ est utilisé comme distance pour construire le dendrogramme. La méthode d'alignement multiple des structures est dérivée de la méthode de Rossman et Argos (Rossmann and Argos, 1976) qui est une méthode itérative de superposition-alignement. La fonction de probabilité a été adaptée : la probabilité que les résidus i de la protéine A et j de la protéine B soient en équivalence dépend à la fois de la distance entre les C_α des résidus i et j et des distances entre les C_α des résidus $(i-1, j-1)$ et $((i+1, j+1))$. La programmation dynamique (locale, de type Smith et Waterman) permet ensuite de trouver le meilleur alignement. Les paires de résidus ayant une probabilité trop faible sont éliminées et les deux structures sont superposées selon la nouvelle équivalence. Les nouvelles superpositions sont calculées à partir de coordonnées moyennes d'un ensemble de C_α .

Beaucoup de méthodes de comparaison de deux structures ont été adaptées pour la comparaison multiple en utilisant le même type de méthode. Par exemple, SSAP (page 51) a été adapté (Taylor et al., 1994) à l'alignement multiple en le combinant avec l'algorithme d'alignement multiple de séquences MULTAL (Taylor, 1988). La représentation des structures est ici faite par les vecteurs définis dans SSAP et un vecteur moyen est utilisé pour les alignements temporaires.

Une autre méthode plus récente, MAMMOTH (Ortiz et al., 2002), a aussi été adaptée pour l'alignement multiple avec construction d'un dendrogramme (Lupyan et al., 2005). L'alignement des structures est fait par programmation dynamique (SW) et il est affiné après une étape de superposition des structures par optimisation avec la méthode SIMPLEX.

Le programme MALECON (Ochagavia and Wodak, 2004) travaille à partir des alignements de paires de structures trouvés par la méthode de N. Boutonnet (Boutonnet et al., 1995) (voir méthodes par paires). Les résidus alignés dans toutes les structures sont ensuite recherchés pour chaque triplet de structures (voir figure 3.1). Si un nombre suffisant de résidus est trouvé, les structures sont superposées deux à deux et la structure qui possède les $RMSD_c$ les plus faibles avec toutes les autres est la structure pivot. Les résidus spatialement proches sont alors ajoutés à l'alignement et un nouvel alignement est défini. Les autres structures sont ensuite ajoutées une par une à cet alignement de trois structures : un résidu d'une nouvelle protéine est ajouté à l'alignement si il est aligné avec tous les résidus présents dans les alignements de paires de structures. Bien qu'aucun dendrogramme ne soit construit, les structures sont donc ajoutées progressivement à l'alignement multiple.

La méthode CE-MC (Guda et al., 2001; Guda et al., 2004) utilise le programme CE (page 45) comme première étape. Toutes les paires de structures sont alignées et les Z-scores sont utilisés pour construire un arbre selon la méthode UPGMA (Sneath and Sokal, 1973). Les structures sont ensuite

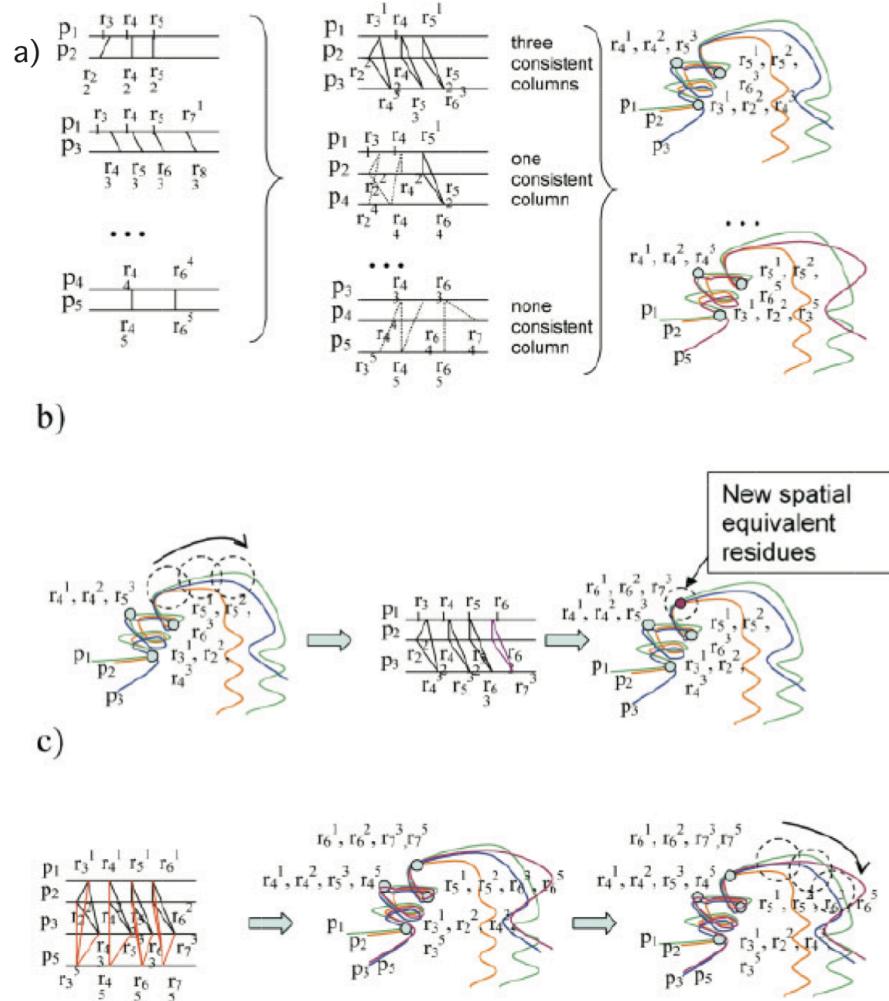


FIG. 3.1 – Figure extraite de l'article de Ochagavia et Wodak (Ochagavia and Wodak, 2004). Cinq structures sont alignées : p_1, p_2, p_3, p_4 et p_5 .

- a) Gauche : les alignements des paires de structures permettent d'établir les équivalences entre résidus r_x .
- Milieu et droite : si les trois mêmes résidus sont alignés dans trois structures, ils peuvent permettre de créer un alignement multiple (*consistent column*).
- b) Extension de l'alignement multiple : les trois structures sont superposées, l'une des structures étant choisie comme pivot. Les résidus proches spatialement sont ajoutés à l'alignement.
- c) Une quatrième structure est ajoutée si ses résidus sont alignés avec chacun des résidus présents dans l'alignement. L'alignement est étendu à nouveau comme en (b).

La méthode d'alignement du programme récent MATRAS (Kawabata, 2003) est assez similaire. La similarité entre toutes les paires de structures est calculée par alignement (voir section sur l'alignement des SSE, page 60) et un arbre est construit par la méthode UPGMA. L'alignement entre deux groupes suit celui entre les deux structures appartenant à chacun des groupes qui alignent le mieux.

POSA (Ye and Godzik, 2003) est une adaptation de FATCAT (page 47) à l’alignement structural multiple. Un alignement multiple est représenté par un graphe acyclique partiellement ordonné, les nœuds sont les résidus alignés et sont ordonnés selon leur séquence, les résidus non alignés forment les arêtes (ces résidus sont partiellement ordonnés). L’alignement - structure/structure ou structure/groupe ou groupe/groupe - est progressif selon un arbre guide calculé par simple lien sur la matrice de distance entre structures (les alignements deux à deux préalables sont effectués avec FATCAT). De même que l’alignement deux à deux, l’alignement aux étapes supérieures (graphes avec régions non ordonnées) se fait par programmation dynamique sur les AFP (page 47).

3.2.3 Méthodes avec structure pivot ou moyenne

La méthode proposée par Gertsein et Altman (Gerstein and Altman, 1995) a été développée pour trouver un cœur structural d’une famille de N structures. Un alignement multiple ou cœur est donc composé non pas de blocs structuraux, mais de résidus en équivalence pouvant être discontinus, une équivalence contient alors un résidu de chaque structure. Un alignement multiple initial étant donné, les deux étapes suivantes sont répétées : i) calculer la position moyenne de tous les éléments de ce cœur et superposer toutes les structures par rapport à cette structure moyenne, ii) calculer la variation des positions de toutes les structures par rapport à la structure moyenne et ôter la position ayant la plus forte variation. Ces deux étapes sont donc répétées jusqu’à ce qu’il n’y ait plus de résidu dans le cœur. L’analyse statistique des résultats de cette procédure permet d’identifier le cœur.

L’obtention de la structure moyenne de l’étape (i) est itératif : chacune des N structures est prise comme structure de référence, les $N - 1$ structures restantes lui sont superposées et une structure moyenne est calculée. Ces N structures moyennes sont ensuite superposées en minimisant l’équation 3.1 (page 69) avec des poids unitaires. Si le résultat E est inférieur à un seuil donné, une des structures est choisie comme structure moyenne pour l’étape suivante (ii) sinon, le processus est réitéré avec les N structures moyennes à la place des N structures réelles. La variation structurale à chaque position du cœur est définie par les valeurs propres ($\sigma_x^2, \sigma_y^2, \sigma_z^2$) de la matrice des variances/covariances des coordonnées à chaque position. Ces valeurs propres sont ensuite considérées comme les axes d’une ellipse : $V = \frac{4}{3}\pi\sqrt{\sigma_x^2\sigma_y^2\sigma_z^2}$, $R = \frac{1}{9}\pi\sqrt{\sigma_x^2 + \sigma_y^2 + \sigma_z^2}$. La mesure de variation structurale à chaque position est le volume de l’ellipse.

Les éléments sont donc retirés les uns après les autres, les éléments ayant la plus grande variabilité structurale d’abord. Pour décider à partir de quel moment les éléments restant sont les éléments du cœur, la variance des volumes des ellipses des éléments hors cœur est calculée à chaque itération de la méthode. Le seuil est ensuite placé au point où cette variance chute brutalement : les éléments du cœur sont ceux qui n’ont pas été encore éliminés à ce point.

La méthode décrite par M. Gerstein et M. Levitt (Gerstein and Levitt, 1996; Gerstein and Levitt, 1998) est beaucoup plus simple : toutes les paires de structures sont superposées et alignées par une méthode itérative classique de superposition-alignement par programmation dynamique dérivée de la méthode ALIGN (Cohen, 1997). Toutes les structures sont superposées avec une structure « mé-

diane » choisie (celle qui est en moyenne la plus proche des autres structures). Après superposition, de nouveaux alignements entre la structure médiane et les autres structures sont calculés par programmation dynamique, le score dépendant de la distance externe entre les C_α . Ces alignements sont ensuite combinés pour former l'alignement multiple et les structures sont à nouveau superposées selon ces nouvelles correspondances entre C_α . La méthode développée par Wu et coll. (Wu et al., 1998b; Wu et al., 1998a) se déroule selon un processus similaire mais au lieu d'une structure médiane, une structure moyenne est calculée.

Dans la méthode de J. Ye (Ye and Janardan, 2004), le score de l'alignement dépend du *URMS* défini par L. Chew (Chew et al., 1999) et décrit page 33. La structure pivot choisie est celle minimisant la somme de toutes les distances entre le pivot et les autres structures. Une fois les structures alignées sur le pivot, la procédure est itérée avec un nouveau pivot jusqu'à convergence.

La méthode ESCAN (Escalier et al., 1998) (page 28) qui permet de comparer deux structures au niveau atomique peut aussi comparer plusieurs structures si l'une d'entre elles est choisie comme pivot. Il est aussi possible de choisir comme pivot un motif structural prédéfini.

Le programme MUSTA (Leibowitz et al., 1999; Leibowitz et al., 2001) extrait les k-uplets des structures sources dans une table de hachage, en utilisant leurs distances internes et leur géométrie comme clef de hachage (voir explication du hachage géométrique page 30). De même, les k-uplets sont extraits de la structure de référence et servent à interroger la table de hachage, ce qui donne des paquets de k-uplets. Les paquets contenant toutes les structures sources passent ensuite à l'étape d'assemblage en « prototypes » (qui sont faits par paire de structures).

Haraldson et Ohlson (Haraldsson and Ohlsson, 2004) développent une approche floue pour l'alignement multiple de structure. L'idée est d'aligner itérativement et de manière floue chacune des structures avec une structure consensus commune (l'alignement joue sur les C_α). La procédure suivante - en 3 étapes - est itérée : i) alignement flou par paire de chacune des structures avec le consensus, ii) rotation-translation pondérée pour chacune des structures, iii) calcul d'un nouveau consensus.

(i) L'alignement flou est réalisé par une méthode similaire à la programmation dynamique mais avec deux différences importantes. Premièrement, le score est un coût, D_{ij} - qui dépend de la distance entre les atomes i et j et du nombre de gap et de leur localisation - et ce score change tout au long de la procédure d'alignement. Deuxièmement, le chemin proprement dit dans la matrice d'alignement est flou, c'est-à-dire qu'on ne choisit pas le chemin D_{ij} comme étant le meilleur des trois chemins préalables, $D_{i-1,j}$, $D_{i-1,j-1}$ ou $D_{i,j-1}$, additionné du coût du pas à faire, mais un chemin flou, somme pondérée des trois chemins. Chaque chemin est pondéré par un facteur de type Boltzmann dépendant de son coût et d'une « température » (comme dans les procédures de recuit simulé, la température descend au cours des itérations successives). De cette matrice d'alignement flou, des probabilités d'appariement entre chaque couple i et j de C_α de la structure et du consensus sont estimées en multipliant la probabilité que le chemin optimal passe par la case (i, j) par la probabilité d'alignement de la paire i, j . Ceci résulte en une matrice d'appariement floue, W_{ij} , reliant tous les atomes de la structure à tous les atomes du consensus (cette étape est faite pour chacune des structures). Les attributions ne

sont donc pas univoques, mais « réparties ». (ii) La rotation-translation optimale de chaque structure sur le consensus est obtenue en tenant compte de la matrice d’attribution floue, W_{ij} , c’est-à-dire en minimisant :

$$E = \sum_{i=1}^M \sum_{j=1}^M W_{ij} (a + Rx_i - y_i)^2 \quad (3.2)$$

avec a la translation et R la rotation, x_i les atomes de la structure, y_i les atomes du consensus. Un atome d’une structure étant attribué à des degrés divers à plusieurs atomes du consensus (et donc vice-versa), chacune de ces attributions joue donc dans la détermination de la matrice de translation-rotation optimale. (iii) Ayant appliqué chacune des translations-rotations à chacune des molécules considérées, la position de chaque atome du consensus est calculée comme étant la moyenne pondérée des positions de chaque atome des structures auquel cet atome du consensus est « partiellement » attribué. Les alignements structuraux de cette méthode semblent aussi bons sinon meilleurs que ceux obtenus avec d’autres méthodes (notamment pour la méthode d’alignement 2 à 2 que les auteurs ont aussi développée auparavant (Blankenbecler et al., 2003)).

3.3 Méthodes sans alignement des paires

3.3.1 Méthodes utilisant les graphes

L’algorithme de Crandell et Smith (Crandell and Smith, 1983) a été adapté par Brint et Willett (Brint and Willett, 1987) à la comparaison multiple de structures. Pour qu’un ensemble d’atomes - de distances internes similaires - soit conservé, il suffit qu’il soit présent dans toutes les structures. Nous avons vu (page 28) qu’il est aussi possible de trouver les sous-structures communes à deux protéines en construisant un graphe de correspondance dans lequel les cliques maximales sont recherchées. Il est aussi en théorie possible de construire un tel graphe pour plusieurs structures. Une dernière méthode, présentée par Holliday et Willett (Holliday and Willett, 1997), utilise un algorithme génétique, mais elle est plutôt adaptée à la construction de pharmacophores.

La méthode de Koch et coll. (Koch et al., 1992; Koch et al., 1996) travaille au niveau des structures secondaires. Un graphe orienté et étiqueté est défini pour chaque structure, les sommets représentant les hélices α ou feuillets β . Un arc est défini entre deux sommets si les deux SSE sont proches (*i.e.* ont un certain nombre d’atomes à moins d’une distance donnée). Dans ces graphes, les composantes connexes sont appelées des « graphes topologiques ». Pour chercher les sous graphes communs maximaux dans plusieurs graphes topologiques, il faut calculer tous les sous graphes communs maximaux dans toutes les paires de graphes. Pour cela, toutes les paires d’arcs compatibles d’une paire de graphes topologiques sont exprimées dans un « graphe produit ». Une clique maximale de ce graphe « produit » correspond à une sous structure commune maximale dans les deux graphes topologiques. Les cliques maximales de ce « graphe produit » sont recherchées par *branch and bound* (réduction de l’espace : seules sont construites les cliques maximales qui correspondent exclusivement aux sous graphes connectés dans les graphes topologiques).

La méthode de Su et coll. (Cook and Holder, 1994; Su et al., 1999) travaille à la fois au niveau des SSE et des résidus : deux graphes sont construits, un pour chaque niveau. Cette méthode permet plutôt de trouver de petits motifs.

3.3.2 Méthode de hachage géométrique

Le programme MASS (Dror et al., 2003b; Dror et al., 2003a) génère un alignement multiple de SSE. Les auteurs considèrent que si un coeur doit exister, il contient au moins 2 SSE (« 2-SSE »). La stratégie générale est donc de rechercher les paires similaires de 2-SSE dans toutes les structures. Ces paires similaires sont détectées par hachage géométrique. Un SSE est représenté par son type (α, β), son axe, sa longueur et sa direction. Les relations géométriques entre deux SSE sont : l'angle entre leurs axes, la distance entre les milieux des axes, la distance minimum entre leurs axes. En ajoutant le type des SSE, on obtient une grille (espace discrétré) en 5D où l'on peut ranger les 2-SSE, et retrouver par hachage les paires de 2-SSE qui sont similaires. Deux paires (de 2-SSE) similaires sont ensuite rassemblées (en une paire de 4 SSE) si la transformation (translation-rotation) pour aligner une paire en aligne une autre. L'alignement est ensuite étendu au niveau des résidus. Cette méthode est vraiment multiple dans le sens où il n'y a pas de pivot ni référence.

3.3.3 Méthodes de recherche de motifs répétés

La méthode décrite par H. Soldano et coll. puis par M.-F. Sagot et coll. (Soldano et al., 1995; Sagot et al., 1995; Jean et al., 1997), nommée KMRC, est inspirée de l'algorithme de recherche de motifs répétés exacts de Karp, Miller et Rosenberg (Karp et al., 1972), nommé l'algorithme de KMR. Je décrirai en détail cette méthode car une des méthodes développées pendant ma thèse en est dérivée. Le formalisme et les notations sont différents de ceux des articles cités mais sont les mêmes que ceux utilisés pour la description de la nouvelle méthode.

La conformation du squelette d'une protéine peut être décrite par une suite d'angles (ϕ, ψ) ou (α, τ). Si les valeurs des angles sont échantillonnées suivant un pas - ou maille - δ , la conformation du squelette peut donc être décrite par une chaîne de « caractères » d'un alphabet fini comptant $360/\delta$ symboles. La recherche de similarités structurales entre structures se ramène alors à la recherche de motifs répétés dans des chaînes de symboles constituées de la concaténation des structures. L'alphabet sera nommé Σ et le « texte » dans lequel les motifs sont cherchés s . Le symbole en position i dans le texte sera noté $s[i]$. Je nommerai **k -motif** un motif de longueur k , **k -motif répété**, un motif présent au moins une fois dans au moins q structures, q étant le **quorum**⁹. Dans le cas de motifs exacts, une **instance** d'un motif représente ce motif. Par contre, dans le cas de motifs approchés, une instance ne représente pas le motif car elle est « stricte ». Je ferai donc la distinction entre motif et instance d'un motif. Les **occurrences** p d'un motif sont les positions de ses instances. L'ensemble des occurrences d'un motif sera nommé **extension**.

⁹Dans tous les exemples, le quorum est de 2.

La recherche de motifs répétés est un problème connu et très étudié. Beaucoup de solutions ont été proposées dont certaines permettent de trouver des motifs approchés (Crochemore and Rytter, 1994; Sagot, 1996). Dans l'algorithme de KMR, la construction des motifs répétés exacts est progressive : les motifs de taille k permettent de construire les motifs de taille $2k$. En effet, si un motif de taille k a pour extension $\{i, j\}$ et qu'un second motif répété a pour extension $\{i+k, j+k\}$, il est possible de combiner ces deux motifs pour former le motif de taille $2k$ présent en i et j . La recherche des motifs répétés est donc faite en largeur d'abord.

Cependant, les motifs construits à chaque étape ne respectent pas obligatoirement la condition de quorum, ils ne sont pas toujours répétés. Soient par exemple deux k -motifs, un dont l'extension est $\{i, j\}$ et l'autre dont l'extension est $\{i+k, l\}$ avec $l \neq j+k$. La combinaison de ces deux k -motifs aura pour extension $\{i\}$, le nouveau motif n'existant pas en j . Ce nouveau $2k$ -motif n'est pas un motif répété, il doit donc être éliminé. Il n'est pas possible de savoir quel motif sera répété avant de les avoir construits. Une étape de filtrage des motifs non répétés - *i.e.* ne respectant pas le quorum - est donc nécessaire après chaque étape de construction.

Il serait bien sûr dommage de ne construire que des motifs de taille multiple de deux mais il n'est pas obligatoire que deux motifs soient contigus pour être combinés : ils peuvent être chevauchants et donner naissance à des motifs de taille intermédiaire (entre k et $2k$). En théorie, il est aussi possible de combiner des motifs de taille variable. Cependant, il est plus simple qu'à une étape donnée, tous les motifs aient la même taille. Un des intérêts de cet algorithme est qu'à chaque étape, seuls les motifs existants réellement sont créés et traités.

La méthode proposée par H. Soldano et coll. est une généralisation de cet algorithme permettant de trouver des motifs approchés. Pour trouver des motifs approchés, certains symboles sont considérés comme similaires. Soit par exemple l'alphabet $\Sigma = \{a, b, c\}$, a est similaire à b et b est similaire à c (mais a n'est pas similaire à c). L'instance de 2-motif « aa » est similaire à « ab », et « ab » est similaire à « ac » mais « ac » n'est pas similaire à « aa ». Ces similarités peuvent être représentées par un pavage de l'espace des symboles : l'espace de l'alphabet Σ est découpé en **pavés** recouvrants G_i tels que l'ensemble des pavés $G = \{G_1, G_2, \dots, G_{|G|}\}$ avec $G_i \subseteq \Sigma$ pour $1 \leq i \leq |G|$ et aucun $1 \leq i, j \leq |G|$ avec $i \neq j$ tel que $G_i \subseteq G_j$. Ces pavés peuvent donc être chevauchants mais ne peuvent pas se recouvrir totalement. Les symboles dans un même pavé sont similaires. Dans notre exemple, l'alphabet est découpé en deux pavés : $G_0 = \{a, b\}$ et $G_1 = \{b, c\}$. L'alphabet utilisé pour décrire les motifs sera pour les motifs approchés cet ensemble de pavés G_i , aussi nommé **alphabet dégénéré**. L'algorithme de construction des motifs suit le même principe que l'algorithme pour les motifs exacts : les motifs de taille k permettent de construire les motifs de taille $2k$. Il faut cependant noter que - puisqu'un symbole peut appartenir à plusieurs pavés - plusieurs motifs différents peuvent exister en une même position. Toujours avec le même exemple, l'instance « ab » est similaire à « aa » et à « ac » mais ces deux instances sont différentes. L'instance « ab » appartient donc à deux 2-motifs, qui sont les motifs G_0G_0 et G_0G_1 lorsqu'ils sont écrits avec l'alphabet dégénéré.

Un effet de cette dégénérescence est que beaucoup plus de motifs sont générés à chaque étape et

certains des motifs ont une extension incluse dans l'extension d'un autre motif. Prenons par exemple le texte s « $bbba$ », le même alphabet et les mêmes pavés que précédemment. Le symbole b appartient à plusieurs pavés, le texte peut donc être représenté de la manière suivante :

<i>positions</i>	1	2	3	4
<i>symboles</i>	a	b	b	a
<i>pavés</i>	G_0	G_0	G_0	G_0
		G_1	G_1	

Quatre 2-motifs sont présents : G_0G_0 dont l'extension est $\{1, 2, 3\}$, G_1G_0 dont l'extension est $\{2, 3\}$, G_0G_1 dont l'extension est $\{1, 2\}$, et G_1G_1 dont l'extension est $\{1\}$. Ce dernier motif ne respecte pas le quorum, il est donc éliminé. Par ailleurs, les deux motifs G_0G_1 et G_1G_0 ont chacun une extension contenant deux occurrences, mais ces extensions sont chacune incluse dans l'extension du premier motif G_0G_0 . Ces deux motifs sont donc éliminés et seul le premier motif, G_0G_0 , est conservé. Ce motif est dit **maximal**. Les motifs maximaux seuls sont suffisants pour construire tous les motifs maximaux de taille supérieure.

Cette méthode a été appliquée aux structures protéiques. Les symboles sont des angles discrétilisés selon une « maille » δ donnée. Les symboles sont alors des entiers représentant un intervalle d'angles. Cependant, si $\delta = 10^\circ$, un angle de 8° est représenté par le symbole « 0 » car il fait partie de l'intervalle $[0^\circ, 10^\circ]$, il est alors considéré comme différent de l'angle 11° car ce dernier est représenté par le symbole « 1 » (intervalle $[10^\circ, 20^\circ]$). Or, ces deux angles devraient être similaires. La similarité pour les angles fait intervenir un paramètre entier κ appelé « marge ». Si $\kappa = 1$, le symbole 1 est similaire aux symboles 0 et 2. Deux angles sont donc considérés comme similaires si leur différence angulaire est inférieure à $(2 \times \kappa + 1) \times \delta$.

La complexité de cet algorithme est : $O(n \cdot k_{max} \cdot g^{2k_{max}} \cdot \log k_{max})$ avec n la longueur totale des structures, k_{max} la longueur maximale des motifs répétés, g la mesure de la dégénérescence ou nombre maximal de pavés auxquels un symbole de l'alphabet peut appartenir. Dans l'exemple précédent, la dégénérescence est de 3. Par exemple, le symbole 2 appartient aux pavés $G_0 = \{0, 1, 2\}$, $G_1 = \{1, 2, 3\}$, $G_2 = \{2, 3, 4\}$.

Cette méthode a été appliquée à la recherche de blocs structuraux similaires dans les cytochromes P450 par P. Jean et coll. (Jean et al., 1997) en vue de la modélisation de la structure du P450eryF.

CHAPITRE 4

Comparaison des méthodes décrites

Plusieurs études ont comparé les résultats des différentes méthodes de comparaison structurales. Ces études essaient de répondre à plusieurs questions : une paire de structures se ressemblent-elles par hasard ? Sinon, deux hypothèses peuvent être envisagée : soit elles ont un ancêtre commun, soit elles ont convergé vers une structure proche ? Comment mesurer la qualité d'un alignement structural ? Répondre à ces questions est difficile car la définition de l'alignement optimal de deux structures est variable suivant les méthodes utilisées. En effet, nous avons vu que les méthodes de comparaison de structures utilisent différents modes de représentation des structures. Comme les propriétés comparées ne sont pas les mêmes, les alignements¹⁰ diffèrent. En l'absence de modèle d'évolution pour les structures - comme celui de Dayhoff (Dayhoff et al., 1978) pour les séquences, l'alignement optimal de deux structures n'est pas défini (Godzik, 1996; Feng and Sippl, 1996; Thornton et al., 1995).

4.1 La qualité des alignements

4.1.1 Comparaison manuelle d'alignement

L'objectif de l'étude faite par A. Godzik (Godzik, 1996) était de comparer les alignements structuraux obtenus avec plusieurs méthodes de comparaison structurale dont DALI (Holm and Sander, 1993), SSAP (Taylor and Orengo, 1989b), MNYFIT (Sutcliffe et al., 1987), et GA_FIT (May and Johnson, 1995)¹¹. Trois paires de structures ont été alignées avec les différentes méthodes, la longueur des alignements, le $RMSD_c$ et les résidus mis en correspondance ont été comparés. La première paire de structures alignées était composée de deux protéines homologues liant le cuivre (azurin 1azc chaîne A et plastocyanin 1plc) dont la similarité de séquence est assez faible. Les structures de la seconde paire ont quant à elles des fonctions assez différentes (flavodoxin 4fx2 and protéine CheY 3chy) et une identité de séquence inférieure à 15%. Leur topologie est cependant assez similaire (*Rossmann fold*) ce qui pourrait provenir d'un phénomène de convergence. Les deux dernières protéines font partie de la famille de globines (phycocyanine 1cpcA et myoglobine 1mbc) mais sont cependant très éloignées en séquence et en structure.

Plusieurs observations ont été faites en étudiant et en comparant les différents alignements. D'abord, les $RMSD_c$ des différents alignements varient beaucoup mais ils dépendent fortement de la longueur des alignements. Cependant, deux alignements différents peuvent avoir le même $RMSD_c$ et un alignement légèrement plus long qu'un autre peut pourtant avoir un $RMSD_c$ plus faible. La seconde observation est que les deux programmes GA_FIT et MNYFIT obtiennent souvent des alignements assez différents, bien que les $RMSD_c$ soient dans les deux cas assez bas et que les deux méthodes soient des méthodes itératives de superposition-alignement. Ceci corrobore l'observation que ce type de méthodes est très sensible, notamment au point de départ. Enfin, les alignements peuvent varier d'une méthode à l'autre et d'une paire de structures à l'autre. Pour la première paire (les homologues) presque tous les résidus sont alignés de manière identique par toutes les méthodes. Dans la

¹⁰Dans cette partie, le terme d'alignement signifie à la fois alignement et correspondance.

¹¹Pour ces deux dernières méthodes, voir « Comparaison multiple de structures », page 68

seconde paire (convergence probable), aucun alignement de deux résidus n'est le même dans toutes les méthodes. Pour la troisième paire de structures, les correspondances ne sont les mêmes que pour environ 1/3 des résidus alignés, beaucoup de décalages d'un résidu étant observé pour le reste¹². Les méthodes différant moins sur des structures très éloignées que sur des structures « convergentes », l'auteur en conclut qu'il ne faut pas utiliser les alignements structuraux comme références pour inférer la qualité des alignements de séquences biologiques.

Cependant, trois alignements ne suffisent pas pour tirer une conclusion aussi générale. De plus, il peut y avoir des explications aux différences observées. D'abord, les alignements sont assez similaires pour les deux premières protéines, ce qui est rassurant quand à l'alignement de protéines homologues dont les structures ne sont pas trop éloignées. Ensuite, les deux dernières structures qui sont sûrement homologues mais de toute façon éloignées ont des alignements variables. Ceci peut être dû au fait que les deux méthodes GA_FIT et MNYFIT sont des méthodes d'alignement globales, incapables de fournir un alignement correct lorsque seuls certains blocs structuraux sont conservés comme c'est le cas pour ces deux structures de la famille des globines. Enfin, l'alignement structural résidu par résidu de deux structures non homologues a-t-il un sens ? En effet, aucun ancêtre commun ne lie ces résidus, ne peut-on pas imaginer alors qu'un environnement physico-chimique soit assuré par un résidu dans un repliement et par plusieurs résidus non contigus dans l'autre ? Il ne semble donc pas si inquiétant que les méthodes trouvent des alignements différents pour des structures non homologues - du moment qu'on le sait ! Il est bien sûr intéressant de comparer des structures non homologues mais le niveau de comparaison et la méthode doivent être adaptés. Dans ces cas, une comparaison uniquement au niveau des structures secondaires donnerait peut-être de meilleurs résultats. J'ai comparé manuellement plusieurs alignements structuraux de structures homologues mais lointaines, et les observations confirment celles faites pour la première paire de structure. J'ai notamment comparé les alignements obtenus par DALI (Holm and Sander, 1993), VAST (Gibrat et al., 1996) et CE (Shindyalov and Bourne, 1998) et bien que les méthodes soient très différentes, les paires de résidus alignés sont très souvent les mêmes avec ces trois méthodes. Par exemple, pour l'alignement des protéines 1ATR (protéine chaperonne de mammifère) et 1ATN (deoxyribonucléase de mammifère), les alignements sont de 277 résidus en moyenne, avec 196 résidus alignés de manière équivalente. Il faudrait donc une étude plus poussée pour comparer les comportements des programmes face à une homologie de plus en plus lointaine ou une similarité structurale due à la convergence.

4.1.2 Validation de la qualité d'un alignement par le calcul d'un score.

Beaucoup de scores ont été inventés pour valider la qualité des modèles de prédiction comme le GDT (*Global Distance Test*) de A. Zelma (Cristobal et al., 2001) utilisé pour CASP4. Ils ne sont cependant pas toujours adaptés aux comparaisons de structures différentes, simplement parce qu'ils sont prévus pour comparer une structure et son modèle, et non deux structures différentes. Il est

¹²Exemple de décalage de 1 résidu : dans un premier alignement les résidus a_1 à a_9 de la protéine A sont alignés avec les résidus b_1 à b_9 de la protéine B tandis que dans le second alignement l'alignement est a_1, \dots, a_9 avec b_2, \dots, b_{10} .

cependant utile de savoir si un alignement structural est de bonne qualité. Normalement, les scores finaux calculés l'indiquent mais ces scores sont assez variés et il n'est pas possible de les comparer. Les scores principaux (les plus utilisés) d'après P. Koehl sont (Koehl, 2001) :

- le $RMSD_c$ (équation 2.1 32) ;
- le $RMSD_d$ (équation 2.2 32) ;
- le score de Levitt et Gerstein (Levitt and Gerstein, 1998) (équation 2.7 36) ;
- le score PSD (Protein Structural Distance) de A. Yang et B. Honig (Yang and Honig, 2000a), (équation 2.15 60).

J'ajouterais cependant à ces scores le score élastique de DALI (équation 2.9 43) qui a aussi été utilisé par d'autres méthodes. Aucune étude de ces différents scores n'a été faite mais il serait intéressant d'étudier leurs distributions pour des alignements obtenus avec des méthodes différentes et pour des protéines plus ou moins similaires.

A. C. May (May, 1999) a testé plus de 30 scores de similarités de structure dont le $RMSD_c$ et la similarité de séquence, en se basant sur les familles structurales de HOMALDB (Overington et al., 1993; Sali and Overington, 1994) et les cœurs définis avec la méthode de M. Gerstein et R. Altman (Gerstein and Altman, 1995). Des arbres sont construits pour chaque famille à partir des différents scores calculés pour chaque paire de structures et la cohérence des arbres obtenus avec les scores calculés est testée. Il ressort de cette étude que le $RMSD_c$ est parmi les mesures les plus adéquate pour construire un arbre cohérent.

Pour R. Kolodny et coll. un bon alignement est un alignement le plus long possible et dont le $RMSD_c$ après superposition des structures est faible (Kolodny et al., 2005). Pour comparer les alignements des six programmes étudiés (SSAP, STRUCTAL, DALI, LSQMAN, CE, SSM), plusieurs scores ont été définis, dépendant tous du $RMSD_c$ et de la longueur de l'alignement (L_{ali}). Le plus utilisé est le score $SAS = (RMSD_c \times 100)/L_{ali}$. Un grand nombre de protéines ont été comparées avec chacune des six méthodes (2930 protéines soit plus de 4,2 millions de comparaisons). Les protéines appartenant à la même classe T (Topology) de CATH (Orengo et al., 1997) sont considérées comme similaires. Les différents scores ont été calculés et la capacité des programmes à obtenir de bons scores pour des structures similaires vérifiée. D'après ces scores, les meilleurs alignements sont obtenus par (dans l'ordre) STRUCTAL, SSM et LSQMAN, méthodes optimisant toutes le $RMSD_c$. Pour toutes les méthodes, les meilleurs alignements sont obtenus pour les protéines tout α ou $\alpha - \beta$. Les alignements des structures des classes tout β ont très souvent un score SAS bien supérieur (*i.e.* plus médiocre).

L'utilisation de tels scores basés sur le $RMSD_c$ et la longueur de l'alignement pour évaluer les alignements structuraux introduit cependant un biais car elle favorise les méthodes minimisant le $RMSD_c$ pendant la recherche de l'alignement. De plus, deux structures appartenant à la même classe T de CATH peuvent cependant avoir une structure assez différente (c'est par exemple le cas des 3 paires de structures étudiées par A. Godzik). Dans le cas de structures assez peu similaires, il est difficile de trouver un alignement global, mais il est possible de se limiter à une région structuralement conservée pour obtenir un bon $RMSD_c$. Des méthodes moins globales comme DALI peuvent cependant trouver

plusieurs régions montrant chacune un bon $RMSD_c$ (*i.e.* faible), mais dont la superposition simultanée résulte en un fort $RMSD_c$. De tels alignements ont donc un mauvais score SAS (*idem* pour les autres scores), alors qu'ils représentent une similarité structurale réelle, bien que non globale. Je pense donc que ces scores sont inadaptés à l'évaluation des alignements des méthodes n'utilisant jamais le $RMSD_c$ ou les distances externes. Par contre, c'est une bonne évaluation des méthodes qui l'utilise le $RMSD_c$ et STRUCTAL est sans doute la meilleure des trois méthodes.

Le problème de l'évaluation de la qualité d'un alignement structural reste entier, notamment parce que l'alignement optimal de deux structures n'est pas réellement défini. Il serait cependant possible d'évaluer les méthodes non pas en calculant un score mais en comparant les alignements à ceux vérifiés et modifiés par des experts ou à ceux définis par alignement multiple des structures d'une même famille.

4.2 La sensibilité des méthodes

Le problème d'évaluer la sensibilité et la spécificité des méthodes de comparaison structurale est un peu moins épiqueux. Il faut cependant avoir un ensemble de structures pour lesquelles les liens d'homologie/similarité sont établis. L'identification de ces liens lorsque la similarité de séquences est faible peut alors poser problème, l'expertise humaine est alors nécessaire. Le plus souvent cet ensemble est construit à partir d'une des deux classifications de structures CATH (Orengo et al., 1997) ou SCOP (Murzin et al., 1995) qui sont - au moins en partie - construites manuellement.

L'étude précédente de R. Kolodny et coll. (Kolodny et al., 2005) a évalué aussi la capacité des méthodes à retrouver les paires de structures de même classe T de CATH parmi toutes les autres. Les méthodes permettant d'identifier le plus de paires avec le moins de faux positifs sont DALI (avec lequel 50% des paires sont identifiées avec 1% de faux positifs) puis STRUCTAL (42% pour 1% de faux positifs). Comme pour l'étude de la qualité des alignements, le score SAS a été calculé. Les structures ont ensuite été classées pour chaque méthode selon ce score et il est intéressant de remarquer que CE, SSM, LSQMAN et SSAP obtiennent de bien meilleurs résultats (par exemple SSAP passe de 9% à 48% de vrais positifs pour 1% de faux positifs). Par contre, DALI et STRUCTAL sont plutôt pénalisés par cette mesure. Il faut cependant noter que pour CE, le score final a été modifié depuis cette étude.

Un autre point intéressant de cette étude concerne plutôt la classification CATH. Dans près de 30% des paires de structures ayant un score SAS inférieur à 3 - quelque soit la méthode d'alignement¹³ - et un alignement d'au moins 50 résidus, les deux structures appartiennent à des classes T différentes.

Plusieurs méthodes de comparaison structurale ont été comparées par M. Sierk et W. Pearson (Sierk and Pearson, 2004) :

- DALI , CE, LSQMAN - qui travaillent au niveau des résidus - ;

¹³Un score SAS inférieur 3 pour un alignement de 50 résidus signifie que le $RMSD_c$ est au maximum de 6 Å. Le $RMSD_c$ peut toutefois augmenter si l'alignement est plus long.

- VAST(Madej et al., 1995; Gibrat et al., 1996) et MATRAS (Kawabata and Nishikawa, 2000) - qui travaillent plutôt au niveau des structures secondaires - ;
- SGM (Rogen and Fain, 2003) et PRIDE (Carugo and Pongor, 2002) qui sont des méthodes permettant de calculer un score de similarités sans alignement ;
- PSI_BLAST (Altschul et al., 1997) et SSEARCH (Smith and Waterman, 1981; Pearson, 1991), deux méthodes de comparaison de séquences.

Les paires de structures similaires ont aussi été extraites de la classification CATH. Pour essayer de pallier les problèmes dus à cette classification, quatre jeux de test ont été construits, les protéines considérées comme similaires étant alternativement de la même classe H (*Homolog*) ou de la même classe T (*Topology*). Chaque structure est comparée à toutes les autres et est nommée structure requête. Un taux d'erreur, l'EPQ (*Error per Query*), est alors calculé tel que $EPQ = \text{nombre total de Faux Positifs}/\text{nombre de requêtes}$ pour différentes proportions de structures similaires identifiées (*Coverage* = nombre de Vrais Positifs / nombre total de paires de structures similaires). Les résultats sont montrés figure 4.1.

D'après ces résultats, la méthode la plus sensible et la plus spécifique est DALI. Cependant, MATRAS est souvent meilleur que DALI pour identifier d'abord les structures de classe H, donc fortement similaires (figure 4.1.A). Par contre, si seules les structures de classe T différentes sont considérées comme des faux positifs, DALI et VAST obtiennent de bien meilleurs résultats (figure 4.1.B). De plus, si les paires de structures de même classe T sont les vrais positifs, ces deux méthodes sont clairement les plus efficaces, DALI étant légèrement meilleur. Ceci signifie que ces deux méthodes permettent aussi d'identifier des paires de structures similaires mais lointaines. Les résultats des autres méthodes varient peu entre les deux premiers jeux de données. Il est à remarquer que SGM est moins performant que les méthodes de recherche de similarités de séquences et que ces dernières méthodes sont les meilleures lorsque l'erreur par structure requête (EPQ) est très faible mais que leurs performances sont les moins élevées ensuite. Pour l'identification des protéines de même classe T mais de classe H différente VAST est souvent la meilleure des méthodes. Comme ses résultats pour l'identification des protéines de même classe H sont moins bons, VAST semble adapté pour identifier des similarités structurales générales/faibles/lointaines, par exemple celles de structures analogues et non homologues. Ceci semble confirmer que de telles structures doivent être comparées à un niveau plus global que celui des résidus. L'étude faite par M. Novotny et coll. (Novotny et al., 2004) est reprise et détaillée dans le chapitre sur YAKUSA (page 98).

4.3 Conclusion

La comparaison de structures est un domaine complexe car il n'y a pas accord sur les mesures de similarité, ni sur les éléments sur lesquels doivent s'exercer ces mesures (C_α , SSE, distances internes...). Quelques études ont essayé d'étayer des mesures de similarités par une statistique, mais il est pour l'instant impossible de générer une structure réaliste d'une manière aléatoire. De plus, la si-

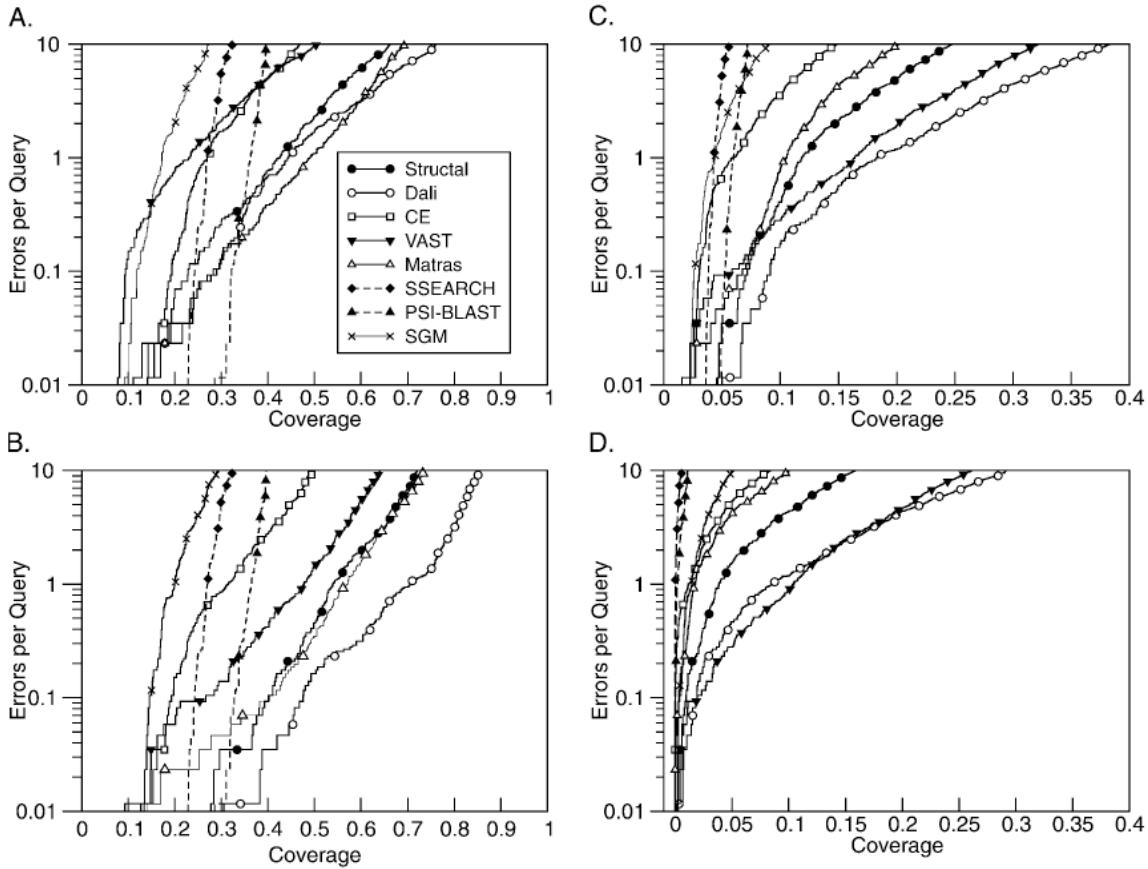


FIG. 4.1 – Erreur par structure requête (EPQ) vs. proportion de paires de structures similaires identifiées (*coverage*). Figure extraite de l’article de M. Sierk et W. Pearson (Sierk and Pearson, 2004).

- A : Les paires de structures similaires (vrais positifs) sont des structures appartenant à la même classe H, les non similaires (faux positifs) toutes les autres paires.
- B : Les vrais positifs sont des structures appartenant à la même classe H, les faux positifs sont des structures n’appartenant pas à la même classe T.
- C : Les vrais positifs sont des structures appartenant à la même classe T, les faux positifs sont des structures n’appartenant pas à la même classe T.
- D : Les vrais positifs sont des structures appartenant à la même classe T mais pas à la même classe H, les faux positifs sont des structures n’appartenant pas à la même classe T.

milarité trouvée dépend du niveau auquel on la cherche (global, local). Comparer deux structures très différentes reste un problème ouvert. Néanmoins, les comparaisons de structure permettent de pousser l’analyse de l’homologie de deux protéines plus loin que ne le peuvent les méthodes de comparaison de séquence à cause de la dégénérescence séquence-structure (Chothia and Gerstein, 1997) et du fait que les structures sont plus « stables » que les séquences.

CHAPITRE 5

Les classifications existantes

L'objectif de cette partie est de présenter les différentes classifications de structures protéiques mais non de présenter toutes les banques de données structurales qui sont très nombreuses. La banque principale de structure est la PDB¹⁴. Cependant, pour naviguer parmi les structures il est souvent plus agréable d'utiliser le site PDBSum¹⁵ qui contient plus d'informations sur les structures et leur séquence ainsi que des liens avec d'autres banques.

Les structures sont généralement catégorisées en quatre classes principales qui ont été définies par M. Levitt et C. Chothia (Levitt and Chothia, 1976) :

- « tout α » ;
- « tout β » ;
- « α/β » (α et β mêlés) ;
- « $\alpha+\beta$ » (α et β organisés en deux régions séparées).

Une des premières classifications - totalement manuelle - a été celle de J. Richardson (Richardson, 1981). Aujourd'hui, les deux classifications les plus connues - et utilisées - sont SCOP (Murzin et al., 1995) (*Structural Classification Of Proteins*) et CATH (Orengo et al., 1997) (pour *Class Architecture Topology Homology*, les quatre premiers niveaux de classification). SCOP et CATH sont toutes deux des classifications hiérarchiques des domaines structuraux protéiques. Les structures sont donc découpées en domaines et sont ensuite regroupées selon leur similarité de séquence, puis leur similarité de structure, puis leur similarité en composition et organisation des structures secondaires. D'autres classifications, cette fois non hiérarchiques, ont été créées avec certaines des méthodes présentées précédemment : MMDB (Gibrat et al., 1996; Madej et al., 1995) (*Molecular Modeling DataBase*) - construite avec le programme VAST et qui est présente sur le site de la PDB - , FSSP (Holm and Sander, 1996b) - construite avec le programme DALI (Holm and Sander, 1993) - et CE Database - construite avec le programme CE (Shindyalov and Bourne, 1998). Ces cinq classifications vont ici être présentées ainsi que quelques autres classifications plus spécialisées.

5.1 Classifications hiérarchiques

5.1.1 SCOP

La classification SCOP (Murzin et al., 1995; Conte et al., 2002; Brenner et al., 1996) est faite manuellement d'après des informations structurales et des connaissances plus générales sur chaque protéine. Les outils automatiques de comparaison structurale ne sont utilisés que pour aider à la classification par inspection visuelle. Les structures protéiques sont tout d'abord découpées en domaines (régions ayant un cœur hydrophobe et peu d'interaction avec le reste de la protéine) puis sont classées.

Les quatre niveaux de classification sont, du niveau le plus général au plus fin :

1. *class* : la composition en structures secondaires est similaire. Il y a quatre classes principales qui sont les mêmes que celles déjà citées et définies par M. Levitt et C. Chothia (Levitt and Chothia,

¹⁴URL <http://www.rcsb.org/pdb/>

¹⁵URL <http://www.ebi.ac.uk/thornton-srv/databases/pdbsum/>

1976). Les 7 autres classes ont un effectif beaucoup plus faible. Les classes des protéines multi-domaines, des protéines membranaires et des petites protéines sont de vraies classes où les protéines ont des caractéristiques spécifiques tandis que les autres sont plus des artefacts dus aux méthodes (classe des protéines ayant une faible résolution, classe des protéines artificielles...) ;

2. *fold* : la composition en structures secondaires (hélices α et feuillets β), leur arrangement spatial et leurs connexions sont similaires ;
3. *superfamily* : l'identité de séquence peut être faible mais où les structures et les fonctions suggèrent une origine évolutive commune ;
4. *family* : les structures protéiques ont au moins 30% d'identité de séquence, ou bien possèdent des fonctions et des structures très similaires.

La banque SCOP est donc une classification « manuelle » de domaines, et la définition des domaines est évidemment critique pour cette classification. La classification SCOP contenait en octobre 2004 945 classes au niveau *fold* et 1539 classes au niveau *superfamily*.

5.1.2 CATH

La classification CATH (Pearl et al., 2005; Pearl et al., 2000; Orengo et al., 1997) est effectuée à la fois automatiquement et manuellement. Comme SCOP, elle est hiérarchique et subdivisée en quatre niveaux principaux. Elle possède trois niveaux supplémentaires de classification établis sur la similarité des séquences protéiques.

Les niveaux de classification sont :

1. **Class** où les structures sont regroupées selon leur composition en structures secondaires et les contacts entre celles-ci. Il y a quatre classes : *mainly α* et *mainly β* qui sont similaires aux deux classes *all α* et *all β* de SCOP, *mixed $\alpha - \beta$* et *Few secondary structures*. L'assignation d'une structure à l'une de ces quatre classes est automatique dans 90% des cas (les 10% restant sont assignés à la main) (Michie et al., 1996) ;
2. **Architecture** où l'organisation générale des structures secondaires est la même pour les structures d'un même groupe. Cette classification est faite manuellement, et notamment par rapport à la classification de J. Richardson (Richardson, 1981) ;
3. **Topology** : où les structures ayant un même repliement en terme de nombre, ordre et connexions de structures secondaires sont regroupées. La méthode de comparaison de deux structures SSAP (Taylor and Orengo, 1989b) est utilisée, avec une contrainte sur la longueur de l'alignement et le score obtenu. ;
4. **Homologous superfamiliy** où les structures d'un même groupe ont des structures et des fonctions très similaires, suggérant un ancêtre commun. SSAP est aussi utilisé ;
5. Les niveaux supplémentaires - et imbriqués - **S,N,I** regroupent les structures ayant une identité de séquence respectivement > 35%, >95% et de 100% (ce dernier niveau regroupe en fait les

protéines qui ont été résolues plusieurs fois, par exemple complexées ou non avec leur ligand). L'algorithme d'alignement des séquences est celui de Needleman et Wunsch.

Les structures sont découpées en domaines structuraux selon le consensus trouvé par trois méthodes indépendantes, DETECTIVE (Swindells, 1995), PUU (Holm and Sander, 1994b) et DOMAK (Siddiqui and Barton, 1995). Si les trois méthodes s'accordent pour le nombre de domaines et si 85% des résidus d'un domaine sont les mêmes, le découpage de DETECTIVE est choisi, sinon, le découpage est fait à la main (47% des cas). Les structures ayant plus de 30 résidus hors domaines sont aussi découpées à la main. Le découpage en domaines structuraux n'est effectué que pour une structure représentative de groupe du niveau N, les protéines du même groupe ayant plus de 95% d'identité héritent des mêmes domaines. La cohérence du découpage en domaines est vérifiée au niveau supérieur S.

Le protocole de classification a un peu varié au cours des années (Pearl et al., 2001). Pour ajouter une nouvelle structure à la classification, cette structure est comparée à la fois au niveau de sa séquence (alignement contre les séquences représentatives et contre des profils PSSM de PSIBLAST) et de sa structure (au niveau peptidique avec SSAP et au niveau des structures secondaires avec GRATH (Harrison et al., 2003)). Des alignements multiples des structures sont réalisés avec la méthode CORA (Orengo, 1999).

L'assignation automatique à une classe (Michie et al., 1996) commence par la détermination des éléments de structure secondaire par SSTRUC qui est une implémentation locale de DSSP (Kabsch and Sander, 1983). Les structures secondaires sont ensuite représentées par des vecteurs et les distances internes entre C_α de deux structures secondaires sont calculés. Le nombre de structures secondaires et les contacts entre structures permet d'établir la classification de la structure dans l'une des quatre classes.

En 2005, CATH contenait 1467 familles au niveau H dont 334 possédaient au moins 3 structures avec moins de 35% d'identité (Pearl et al., 2005) et 813 classes au niveau T .

CATH est donc une classification semi-automatique des structures.

5.2 Classifications non hiérarchiques

5.2.1 FSSP

La classification FSSP (Families of Structurally Similar Proteins (Holm and Sander, 1996c; Holm and Sander, 1994a)) n'est pas hiérarchique. Elle a été construite à partir des alignements de DALI (Holm and Sander, 1993). La méthode est assez simple : elle consiste à découper les structures en domaines structuraux puis à comparer tous les domaines deux à deux avec DALI. Un algorithme de classification ascendante hiérarchique fondé sur les scores des alignements de DALI (*average linkage clustering*) permet ensuite de regrouper les domaines en familles. En réalité, DALI n'est pas utilisé pour comparer directement la structure requête à toutes les structures de la banque. Dans une première étape, la méthode 3DLookUp (Holm and Sander, 1995a), qui compare les structures au niveau de leurs

SSE, est utilisée pour comparer la structure requête avec les structures représentatives des familles de FSSP. Ensuite seulement, DALI est utilisé pour comparer la structure requête avec les structures des familles sélectionnées.

Le problème principal de cette classification est qu'elle n'est pas mise à jour régulièrement et qu'elle est beaucoup moins sophistiquée que les deux classifications précédentes : il n'y a pas de liens avec d'autres banques de données ni de moyen de visualiser les alignements en 3D (seuls les alignements des paires de séquences sont fournis).

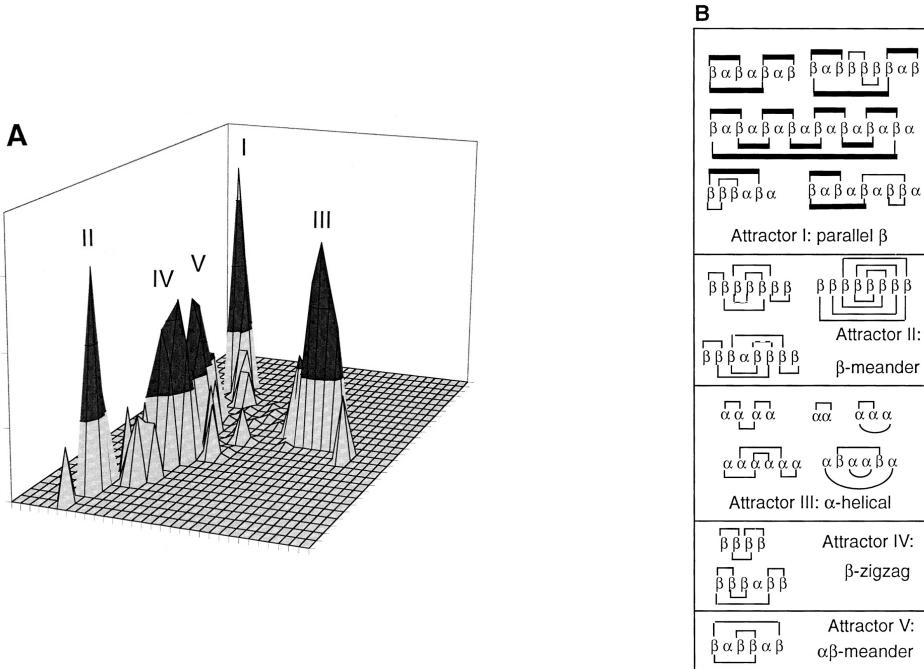
5.2.2 DALI Domain Dictionary

Le dictionnaire de domaine de DALI (Dietmann et al., 2001; Holm and Sander, 1996c) est une classification pleinement automatique des replis protéiques. La taxonomie est dérivée de la similarité de structure, de fonction et de séquence des protéines. Si on considère les protéines comme des points dans un espace à haute dimension - espace des « formes » où les distances structurales entre protéines sont respectées-, Holm et Sander (Holm and Sander, 1996c) ont mis en évidence cinq « attracteurs » pour toutes les structures (figure 5.1(a)). Ces attracteurs servent à classer les domaines de protéines selon leur structure, l'attribution d'un domaine à un attracteur se faisant par un critère de chemin le plus court (deux classes s'ajoutent, l'une pour les structures non « connectées » aux autres, et l'une pour les domaines à égale distance de deux attracteurs). 40% de tous les domaines connus sont couverts par 16 classes de replis (figure 5.1(a)). Plusieurs paramètres raffinent cette première classification structurale : les alignements de séquence de PSIBLAST, l'analyse des regroupements de résidus conservés, la classe enzymatique (*EC number*) et une analyse par mots clés de la fonction biologique.

5.2.3 CE database

La classification est faite en trois étapes :

1. Construction d'une « liste de structures représentatives » en ajoutant une structure à la liste si elle est différente de toutes les représentatives déjà présentes (sinon, on l'ajoute au groupe auquel elle ressemble). Les critères sont le RMS, la différence de longueur, la longueur de l'alignement et le nombre de gap. Les structures sont comparées avec le programme CE (Shindyalov and Bourne, 1998). On obtient des groupes de « premier niveau » ayant souvent une forte similarité de séquence ;
2. Comparaison des structures représentatives des groupes du premier niveau selon les mêmes critères mais plus « lâches » ; il faut que le z-score soit au dessus de 4 ;
3. Recherche des sous structures communes dans ces groupes de second niveau (alignements 2 à 2 mais on cherche les sous-structures les plus chevauchantes).



(a) La quantification des similarités des paires de structures (comparaison « tout contre tout ») donne la position d'une structure dans un espace abstrait de hautes dimensions. La hauteur des pics reflète la densité de population de repliements, les axes horizontaux sont les axes des deux premiers vecteurs propres (*i.e.* associés aux deux plus grandes valeurs propres), l'axe vertical donne le nombre de repliements. La distribution des architectures est donnée par la projection sur le plan (la proximité sur ce plan donne une indication sur la similité structurale entre 2 protéines).

(b) 40% de tous les domaines connus sont couverts par 16 classes de repliements. Ces 16 repliements sont montrés ici sous forme de diagrammes topologiques de structures secondaires dans la classe de leur attracteur (le numéro d'attracteur est le même que dans la figure 5.1(a)).

FIG. 5.1 – Figures extraites de l'article de Holm et Sander (Holm and Sander, 1996c).

5.3 Alignements multiples de familles de structures

5.3.1 3D-PSSM

3D-PSSM est en fait une banque de profils (Kelley et al., 2000; Fischer et al., 1999; Kelley et al., 1999)¹⁶. Elle est surtout utilisée pour la reconnaissance de repliements. Elle se fonde sur les alignements structuraux de SCOP pour établir les équivalences entre résidus. Ces équivalences sont ensuite utilisées pour étendre les alignements multiples de séquences obtenus par des recherches de similités de séquences. Ces alignements étendus sont convertis en matrices de scores-position spécifique (PSSM) qui - combinées avec l'alignement des SSE prédicts par PRIDE et avec les potentiels d'accès-

¹⁶URL : www.sbg.bio.ic.ac.uk/~3dpssm/

sibilité au solvant des résidus tirés de DSSP et normalisés - permettent la reconnaissance structurale et fonctionnelle de séquences au delà des techniques basées sur les seules séquences (PSIBLAST). Les alignements des séquences requêtes sur ces profils sont réalisés par programmation dynamique. 3D-PSSM est l'une des méthodes de *threading* les plus performantes.

5.3.2 HOMSTRAD

HOMSTRAD (Mizuguchi et al., 1998) est une banque d'alignements de structures de familles d'homologues (1032 familles / 3454 structures alignées en 2005). L'homologie est ici inférée par un degré d'identité suffisamment élevé. Les classifications de SCOP, Pfam, PROSITE et SMART sont combinées avec les résultats de recherche de similarités de PSI-BLAST et de FUGUE - une méthode de *threading* liée à HOMSTRAD - pour définir les familles, qui sont représentées de manière à montrer l'environnement structural local de chaque résidu.

5.3.3 PALI

PALI (Gowri et al., 2003; Sujatha et al., 2001; Balaji et al., 2001) est une base d'alignement reposant sur les structures de protéines homologues ; elle contient aussi les arbres phylogénétiques de domaines de protéines homologues (actuellement, 844 familles d'au moins deux membres incluant 3863 domaines, 817 familles constituées d'une structure unique). Les familles de SCOP ont été utilisées comme base pour établir des alignements par paire et multiples avec STAMP (page 71). A chaque famille, sont associés deux arbres calculés avec le programme PHYLIP (Felsenstein, 1985) : l'un basé sur la dissimilarité structurale (SDM) définie sur chaque paire et l'autre sur la similarité de séquence des résidus topologiquement équivalents (le SDM combine le $RMSD_c$ et la longueur de l'alignement (Johnson et al., 1990a; Johnson et al., 1990b)).

5.3.4 LPFC

LFPC (*A Library of Protein Family Cores*)¹⁷ (Schmidt et al., 1997) est une base de cœurs structuraux. Les cœurs sont construits avec la méthode des « ellipsoïdes » de Gerstein et coll. (page 73). Ils constituent des sortes de moyenne pondérée pour chaque famille de structures : ces cœurs contiennent des résidus à faible et forte variation spatiale (les résidus à faible variation spatiale sont les résidus des différents membres de la famille qui se superposent bien). Dans la base, les cœurs sont organisés selon la méthode d'alignement multiple utilisée (HOMALDB, FSSP ou « à la main »), et une famille peut donc avoir plusieurs cœurs.

5.4 Banques d'alignements structuraux

¹⁷URL <http://smi-web.stanford.edu/projects/helix/LPFC/>

5.4.1 HOMALDB

HOMALDB (Overington et al., 1993; Sali and Overington, 1994) est une banque de structures homologues alignées. Les paramètres structuraux utilisés pour l'alignement sont l'accessibilité au solvant, la structure secondaire, les contacts de Van der Waals, les liaisons hydrogènes et les coordonnées (le programme COMPARER a été utilisé, voir page 51). Cette banque contient des profils caractérisant les familles (obtenues par PSIBLAST).

5.4.2 MMDB

MMDB (Molecular Modeling DataBase) est la banque d'alignements de VAST (Gibrat et al., 1996; Madej et al., 1995) (page 56). Les structures sont toujours découpées en domaines structuraux par un algorithme recherchant les points de rupture entre groupes de structures secondaires. Cet algorithme compare les contacts entre les éléments de structures secondaires.

Deuxième partie

Nouvelles méthodes

CHAPITRE 6

Yakusa : une nouvelle méthode de recherche de similarités structurales

6.1 Méthode

Notre objectif est de comparer rapidement une structure protéique dite requête à toutes les structures présentes dans une banque et de trouver dans ces dernières les similarités structurales locales avec la structure requête. Nous utilisons une description linéaire des structures ce qui nous a permis d'appliquer le même type d'algorithme que celui utilisé dans le programme BLAST (Altschul et al., 1990) spécialisé dans la recherche de similarités dans les séquences nucléiques ou protéiques.

6.1.1 Description de la structure tridimensionnelle des protéines

Nous utilisons une représentation en coordonnées internes du squelette protéique : les angles α et τ tels qu'ils ont été décrits par Levitt (Levitt and Warshel, 1975; Levitt, 1976). Cette description est exposée extensivement dans la section « Description géométrique du squelette ».

Pour rappel, l'angle α est l'angle dièdre défini par quatre C_α successifs et l'angle τ est l'angle plan défini par trois C_α successifs (voir figure 1.8). L'angle α_i est associé au deuxième des quatre C_α (le $C_{\alpha i}$) et l'angle τ_i est donc celui formé par les trois premiers C_α . Comme nous l'avons vu, l'angle τ varie peu autour de 106° (*cf.* équation 1.2 page 16) et l'angle α suffit pour décrire le squelette carboné.

Nous avons vérifié que la reconstruction d'un squelette carboné uniquement à partir des angles α discrétisés ont un $RMSD_c$ correct après superposition avec le fragment initial. Ces valeurs ont été comparées à celles obtenus par superposition de fragment tirés au hasard (voir tableau 6.1 et graphique 6.1).

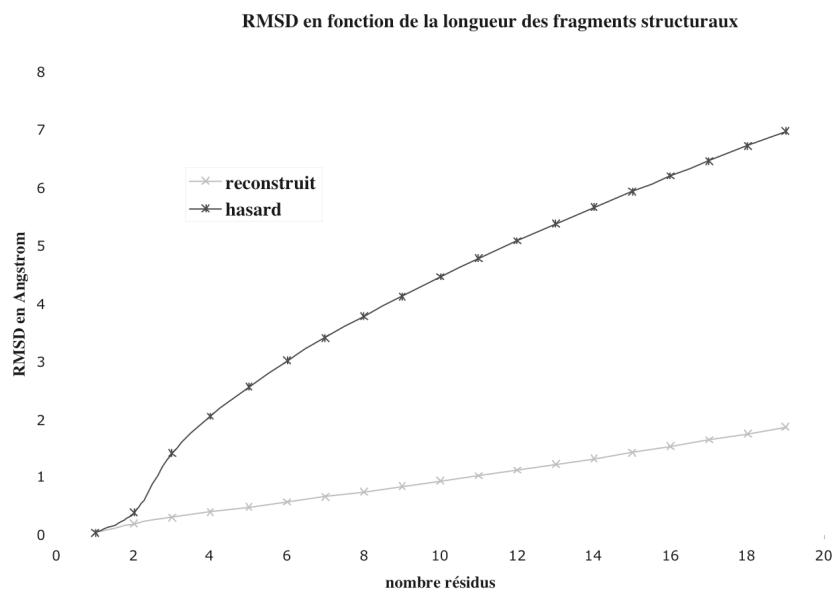


FIG. 6.1 – $RMSD_c$ calculés soit entre tous les fragments structuraux d'une taille donnée de la PDB non redondante et ceux reconstruits à partir de leurs angles α soit entre deux fragments pris au hasard dans la PDB.

D'autre part, et de manière surprenante, à partir d'environ 12 résidus, deux sous-structures pro-

tiques présentant des angles α très grossièrement proches, par exemple à 60° près, ou même plus, se superposent bien dans l'espace des coordonnées cartésiennes. Ceci est dû aux contraintes de conformation des protéines et à l'espace limité des angles α (voir section « Description géométrique du squelette »).

Nous n'utilisons donc que les angles α pour décrire les protéines et nous définissons un angle pour chaque C_α (sauf le premier et les deux derniers). Les angles α sont discrétisés en classes selon une maille. Nous utilisons généralement une maille de 10° , il y a donc 36 classes, chacune représentée par un symbole (un nombre entier). Une structure est alors une suite de symboles sur laquelle les algorithmes de recherche de motifs « classiques » peuvent être appliqués.

TAB. 6.1 – Moyennes et écart-types des $RMSD_c$ calculés soit entre tous les fragments structuraux d'une taille donnée de la PDB non redondante et ceux reconstruits à partir de leurs angles α soit entre deux fragments pris au hasard dans la PDB.

Longueur	Reconstruit		Hasard	
	$RMSD_c$ moyen	σ	$RMSD_c$ moyen	σ
2	0,02	0,24	0,02	0,24
3	0,18	0,36	0,37	0,39
4	0,29	0,42	1,39	0,70
5	0,38	0,47	2,03	0,73
6	0,47	0,52	2,54	0,77
7	0,56	0,57	2,99	0,84
8	0,64	0,62	3,39	0,91
9	0,73	0,67	3,76	0,98
10	0,82	0,72	4,11	1,06
11	0,91	0,76	4,44	1,13
12	1,01	0,81	4,75	1,20
13	1,11	0,86	5,06	1,27
14	1,20	0,91	5,35	1,34
15	1,31	0,96	5,63	1,40
16	1,41	1,01	5,91	1,46
17	1,52	1,06	6,18	1,52
18	1,62	1,11	6,44	1,58
19	1,74	1,16	6,70	1,63
20	1,85	1,21	6,95	1,68

6.1.2 Description générale de l'algorithme

Le principe général de la méthode est d'abord de rechercher tous les petits motifs de taille fixe k communs à la structure requête et aux structures de la banque. La suite de symboles de la structure requête - que nous nommerons S_r - est découpée en motifs chevauchants de taille fixe k qui sont rangés dans un automate, avec leur **position** dans la structure (numéro du résidu). Cet automate contient non

seulement tous les motifs présents dans la requête, mais aussi tous ceux - non présents sur la requête - qui leur sont proches *i.e.* ceux dont les symboles représentent des angles voisins. La structure de l'automate est similaire à celle décrite par A. Aho et H. Corasick pour la recherche en parallèle de plusieurs motifs (Aho and Corasick, 1975).

Après construction de l'automate, la recherche de similarités structurales locales entre la structure requête et chaque structure de la banque se déroule en trois étapes :

première étape : recherche des graines. La structure de la banque est parcourue avec l'automate et les petits motifs communs aux deux structures (**graines**) sont trouvés ;

seconde étape : sélection et extension des graines. Les graines sont sélectionnées et étendues en segments structuraux les plus longs possibles : les **SHSP** (*Structural High Scoring Pairs*) ; la comparaison est faite au niveau des angles ;

troisième étape : sélection des SHSP et calcul d'un score global. Les SHSP compatibles sont sélectionnés et un score global permettant de classer les structures de la banque selon leur similarité structurale avec la requête est calculé.

Chacune de ces étapes est répétée pour chaque structure de la banque.

6.1.3 L'automate

1) Description de l'automate et quelques définitions

Comme dans la méthode Aho-Corasick (qui est une généralisation de l'algorithme KMP (Knuth et al., 1977) pour plusieurs motifs), nous utilisons un automate déterministe fini qui permet la recherche de plusieurs motifs en parallèle. Cet automate permet de parcourir un **texte** (les structures de la banque) en cherchant des **motifs** (les motifs de la requête) de la manière la plus économique possible, c'est-à-dire en ne comparant qu'une seule fois chaque symbole du texte à un symbole de l'automate. Ainsi, le temps que prend la recherche croît linéairement avec la longueur du texte c'est-à-dire la taille de la banque.

Pour aider à la compréhension, l'automate, et sa construction, seront présentés d'abord pour des motifs strictes (sans dégénérescence) ; la dégénérescence sera ensuite introduite.

Pour la recherche de motifs de longueur k , l'automate peut être représenté comme un arbre ayant $k + 1$ niveaux de nœuds, de la racine - niveau 0 - jusqu'aux feuilles - niveau k - et k niveaux de transitions (voir figure 6.2(a)). Une **transition** est un lien associé à un symbole, menant d'un nœud à un autre. Une transition est suivie lorsque que l'on se trouve au nœud d'où elle part et que son symbole est rencontré dans le texte. Chaque nœud de l'automate a autant de transitions qu'il y a de symboles dans notre alphabet d'angles α (voir figure 6.2(b)). Dans l'automate, il y a un chemin possible pour tout motif (même s'il n'est pas présent dans la requête). Cependant, les transitions suivies pour un motif inexistant ne sont pas du même type que celles suivies pour un motif existant : les premières sont des « **transitions échec** » tandis que les secondes sont des « **transitions succès** ». Il existe un

troisième et dernier type : les « **transitions fin** » permettant de poursuivre la recherche après avoir trouvé un motif commun.

Dans un premier temps, seules les transitions formant des motifs existants dans la requête (ou similaires à ceux-ci) sont créées. Nous nommerons cet automate incomplet « **arbre** ».

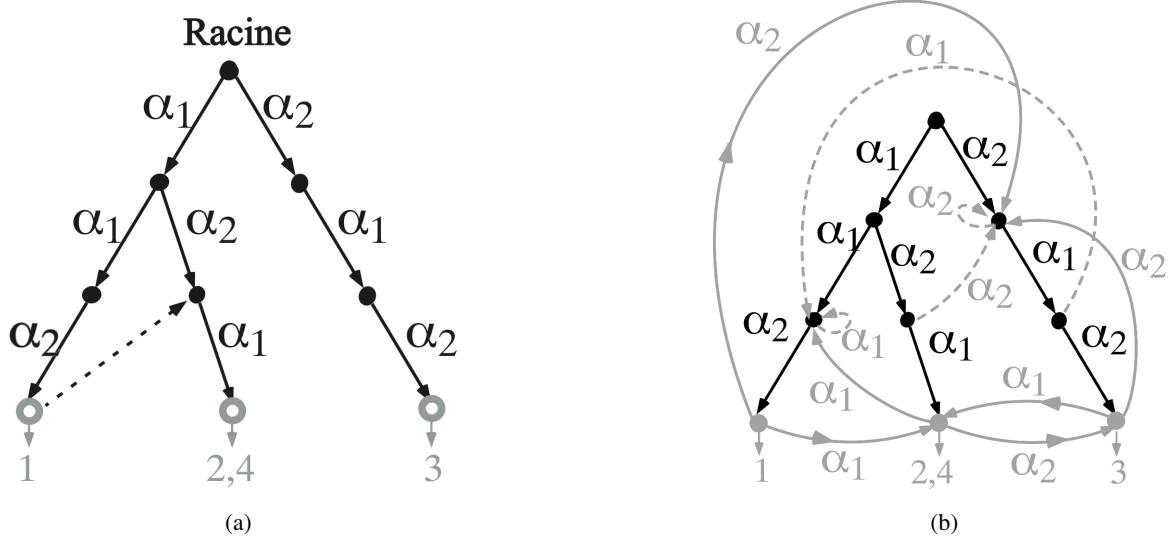


FIG. 6.2 – Automate (sans dégénérescence) sous forme d’arbre pour la structure « $\alpha_1\alpha_1\alpha_2\alpha_1\alpha_2\alpha_1$ » et pour des motifs de taille 3. Pour des raisons de simplicité, l’alphabet ne contient que deux symboles, α_1 and α_2 . Les motifs présents sont $\alpha_1\alpha_1\alpha_2$, $\alpha_1\alpha_2\alpha_1$, $\alpha_2\alpha_1\alpha_2$. Les feuilles (cercles gris et vides) contiennent les positions du premier symbole du motif de la feuille. Les nœuds sont les cercles pleins noirs. Figure 6.2(a) : La flèche pointillée est le lien supplémentaire pour $\alpha_1\alpha_1\alpha_2$; elle mène au motif $\alpha_1\alpha_2$. Les autres liens supplémentaires ne figurent pas pour alléger le schéma mais en réalité il y a un tel lien à chaque nœud ou feuille (sauf la racine), menant toujours à un nœud du niveau immédiatement supérieur (ils sont tous présents dans la figure 6.6).

Figure 6.2(b) : Les trois types de transition sont : les transitions succès (les flèches noires), les transitions échec (les flèches pointillées grises) et les transitions fin (les flèches continues grises).

Si le texte est $\alpha_1\alpha_2\alpha_1\alpha_1\alpha_1\alpha_2$ le premier motif $\alpha_1\alpha_2\alpha_1$ est trouvé en ne suivant que des transitions succès. A partir de la feuille, la recherche est poursuivie avec le symbole α_1 en suivant la transition fin qui lui est associée qui mène au nœud $\alpha_1\alpha_1$. Le symbole encore suivant (α_1) du texte n’a pas de transition succès à ce nœud, c’est donc une transition échec qui est suivie et elle pointe sur le même nœud. Le dernier symbole α_2 existe en dessous du nœud et l’on arrive à la feuille de motif associé $\alpha_1\alpha_1\alpha_2$. Deux motifs sont donc communs à l’automate et au texte : $\alpha_1\alpha_2\alpha_1$ et $\alpha_1\alpha_1\alpha_2$.

Chaque branche « complète » a k symboles associés aux k transitions. Cette suite de symboles forme un motif présent dans la requête (ou similaire à un motif présent). Comme nous ne cherchons que des motifs de taille k , seules les feuilles contiennent les positions des motifs dans la requête. Nous appellerons **motif d’un nœud** (ou d’une feuille) le motif formé par les symboles associés aux transitions qu’il faut parcourir pour atteindre ce nœud en partant de la racine. Les transitions d’un nœud sont celles partant de ce nœud et non pas celles y arrivant. Ainsi, les feuilles n’ont (pour l’instant) pas de transition.

2) La dégénérescence

Afin de détecter non seulement des motifs identiques mais aussi des motifs similaires, un angle α - par exemple - de 20° doit être considéré comme similaire à un angle de 10° ou à un angle de 30° , même s’ils ne font pas partie de la même classe. Pour cela nous introduisons une « **dégénérescence**

locale » δ au niveau de chacun des symboles des motifs. Elle est exprimée en unité de maille, obligatoirement inférieure ou égale à δ_{max} (condition que je nommerai δ_{max}). Par exemple, avec une maille de 10° , le symbole 2, représente les angles α dans l'intervalle $[20^\circ, 30^\circ]$. Avec $\delta_{max} = 1$, il est considéré comme similaire aux symboles 1 et 3 correspondant aux angles α respectivement dans les intervalles $[10^\circ, 20^\circ]$ et $[30^\circ, 40^\circ]$. Cependant, pour éviter l'accumulation et la propagation de telles petites différences angulaires dans les motifs, une seconde condition a été ajoutée : $\sum_{i=1}^k |\delta_i| \leq \Delta_{max}$ (condition que je nommerai Δ_{max}), Δ_{max} étant la « **dégénérescence globale** » aussi exprimée en unité de maille (voir figure 6.3).

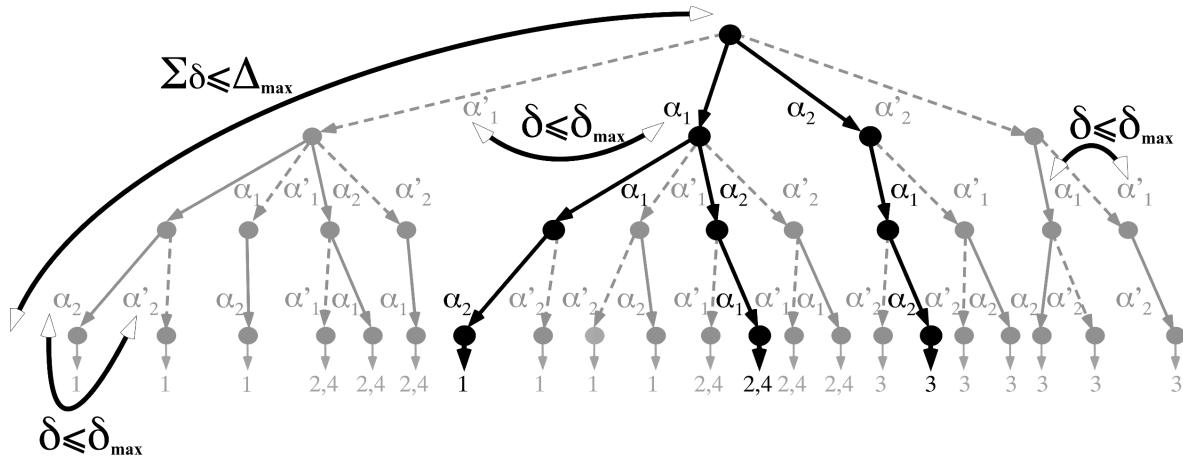


FIG. 6.3 – Arbre avec dégénérescence : la dégénérescence locale δ_{max} est égale à 1 et la dégénérescence globale Δ_{max} à 2. En réalité, un δ_{max} de 1 provoque l'insertion de deux symboles supplémentaires, un pour une dégénérescence positive ($\alpha' = \alpha + 1$) et un pour une dégénérescence négative ($\alpha'' = \alpha - 1$) mais ici seule une des deux dégénérescences est représentée pour simplifier le schéma. L'arbre non dégénéré - « exact » - avait 3 feuilles tandis que celui-ci, malgré la dégénérescence minimale en a déjà 23 (et donc 23 motifs exacts ou similaires). L'arbre « exact » est en noir et les branches portant des symboles dégénérés sont grises. Les flèches pointillées sont associées à des symboles dégénérés tandis que les flèches continues le sont avec des symboles réellement présents. Un Δ_{max} égal à 2 signifie ici qu'il n'y aura pas plus de deux symboles dégénérés de 1 (δ_{max}) par branche.

3) Construction de l'automate

Pour construire de manière efficace l'automate nous avons exploité le fait que tous les motifs de la structure requête S_r sont chevauchants. Comme indiqué ci-dessus, dans un premier temps, cet automate est construit comme un arbre et seuls les motifs existant dans S_r (ou similaires à ceux-ci) sont insérés. C'est pendant cette étape que tous les nœuds sont créés. Dans un second temps, les transitions pour les motifs absents sont créées.

Pour aider à la compréhension de la méthode de construction de l'automate, elle sera d'abord expliquée sans prendre en compte la dégénérescence, puis avec la dégénérescence.

Construction de l'automate exact

Comme tous les motifs de S_r sont chevauchants et de même longueur k , tous les suffixes sont aussi des préfixes (pour un exemple, voir figure 6.2(a)). Il est possible de définir dans l'arbre un « **lien**

suppléant¹⁸, partant d'un nœud dit courant, et arrivant à un « **nœud suppléant** » du niveau immédiatement supérieur. Le motif de ce nœud suppléant est le plus long suffixe du nœud courant, c'est-à-dire le motif du nœud courant sans son premier symbole.

Tous les nœuds ou feuilles ont un nœud suppléant sauf la racine. Pour les fils de la racine, le nœud suppléant est la racine. Lors de la création d'un nœud, son lien suppléant est créé lors de la création du nœud suivant : si le nœud N' est le suppléant du nœud N et si un fils C est ajouté au nœud N , le suppléant de ce nouveau nœud C est le fils C' de N' (voir figure 6.4). Les nœuds des $k - 1$ derniers symboles sont des cas particuliers : ils peuvent ne pas avoir de nœud suppléant au niveau immédiatement supérieur si leur motif n'a jamais été rencontré. Dans le pire des cas, le suppléant est la racine.

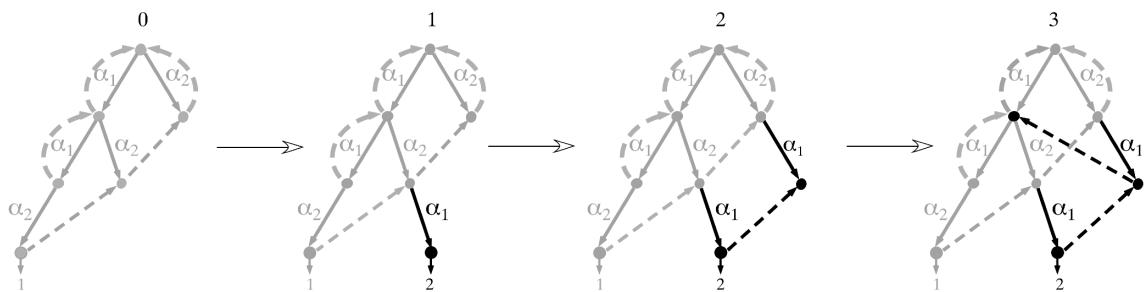


FIG. 6.4 – Insertion du symbole α_1 , quatrième symbole de la structure « $\alpha_1\alpha_1\alpha_2\alpha_1\alpha_2\alpha_1$ » (les flèches pointillées grises sont les liens suppléants.).

0 : Arbre avant l'insertion de α_1 , seuls α_1 , α_1 et α_2 sont présents.

1 : insertion au dernier niveau (k), une feuille est créée (en noir).

2 : insertion au niveau immédiatement supérieur, création d'un nœud et du lien suppléant de la feuille précédente.

3 : insertion au premier niveau, pas de création de nœud car α_1 est déjà présent, création du lien suppléant du nœud précédent.

Le prochain symbole sera inséré dans l'arbre en partant du nœud créé au niveau 2 et les opérations seront les mêmes.

Ces liens suppléants sont utilisés pour la construction de l'automate. Pour chaque symbole (sauf les $k - 1$ premiers) il y a création d'une transition associée au symbole et d'un nœud¹⁹ à chaque niveau de l'arbre, s'ils ne sont pas déjà présents. Ces créations ont lieu dans un certain ordre : en partant du niveau le plus bas jusqu'au niveau en dessous de la racine. Ceci revient à ajouter le symbole dans les motifs en tant que dernière lettre, puis avant-dernière lettre... jusqu'à première lettre (voir figure 6.4). La première transition créée pour le symbole mène donc à une feuille ; le symbole est donc inséré en tant que dernier symbole du motif de la feuille. La position de ce symbole est aussi ajoutée à la feuille. Les autres transitions (et nœuds) pour ce symbole sont ajoutées en dessous des nœuds pointés par les liens suppléants : lorsqu'une transition est insérée en dessous du nœud N la transition suivante pour le même symbole est insérée en dessous du nœud N' , suppléant de N .

L'insertion du prochain symbole se fera en partant de la feuille créée et en suivant son lien suppléant. Les mêmes opérations sont répétées pour chaque symbole : les liens suppléants sont suivis, les tran-

¹⁸Un lien suppléant n'est pas nommé transition car c'est un lien temporaire utilisé pour la construction de l'automate.

¹⁹Comme toute transition pointe sur un nœud ou une feuille, la création d'une nouvelle transition implique forcément la création du nœud sur lequel elle pointe et inversement, du moins pour les transitions associées aux symboles existants ou similaires.

sitions associées aux symboles sont créées si nécessaire ainsi que les liens suppléants des nouveaux nœuds. On peut donc dire que pour chaque symbole une « **liste verticale** » de liens suppléants est parcourue. Il n'est souvent pas nécessaire de remonter jusqu'à la racine car si lors du parcours le symbole est rencontré, il est obligatoirement présent à tous les niveaux supérieurs. Il est alors inutile de continuer à parcourir la **liste verticale**.

Construction de l'automate dégénéré

Lors de la construction de l'automate avec dégénérescence, les motifs dégénérés sont insérés en même temps que les motifs exacts. Autrement dit, les symboles dégénérés selon δ_{max} sont insérés en même temps que les symboles exacts si les conditions δ_{max} et Δ_{max} sont respectées.

Comme plusieurs symboles sont insérés en même temps, l'insertion d'un nouveau symbole ne se fait plus à partir du suppléant d'une feuille mais à partir de tous les suppléants des feuilles créées pour le symbole précédent. Cette liste de points de départ constitue ce qu'on appellera la « **liste horizontale** » de noeuds. En association à cette liste, il faut conserver les dégénérescences locales des motifs de ces nœuds pour pouvoir vérifier que la condition Δ_{max} est respectée.

Ainsi, en partant de chaque nœud de la **liste horizontale**, pour chaque symbole et ses dégénérés et si la condition Δ_{max} est respectée, les opérations sont semblables à celles présentées précédemment : les liens suppléants sont suivis, les transitions associées aux symboles (exact ou dégénérés) créées si nécessaire ainsi que les liens suppléants des nouveaux nœuds. La liste horizontale pour l'insertion du symbole suivant est aussi créée (voir figure 6.5). Cependant, pour un symbole et ses dégénérés, il y a plus de feuilles créées que de nœuds créés au niveau supérieur car les feuilles ont des suppléants en commun. En effet, comme le lien suppléant permet de trouver le motif de la feuille sans le premier symbole, toutes les feuilles dont le motif est identique sauf le premier symbole ont le même suppléant. Et cela est le cas à chaque niveau : tous les nœuds dont les motifs ne diffèrent que par le premier symbole ont le même suppléant. Il est donc souvent inutile de parcourir la totalité de la **liste verticale**. Nous insérons les symboles dans un ordre précis : du moins dégénéré (le symbole exact) au plus dégénéré. Ainsi les motifs des nœuds de la **liste horizontale** sont aussi ordonnés du moins dégénéré au plus dégénéré. Lorsque la **liste horizontale** est parcourue, les premiers nœuds ont des motifs commençant par un symbole exact. Lorsque tous ces nœuds ont été traités, la **liste horizontale** pour le symbole suivant est complète. Pour tous les nœuds restants de la **liste horizontale** courante, il suffit de créer uniquement les feuilles et leurs liens suppléants (si nécessaire). L'algorithme de construction de construction est présenté page 106.

4) Les dernières transitions

Pour que l'arbre soit un réel automate, il faut qu'à chaque nœud il y ait une transition pour chaque symbole de l'alphabet. Ainsi, chaque symbole du **texte** (les structures de la banque) permet d'activer une transition dans l'arbre quel que soit le nœud où l'on se trouve. A ce stade, l'automate ne contient

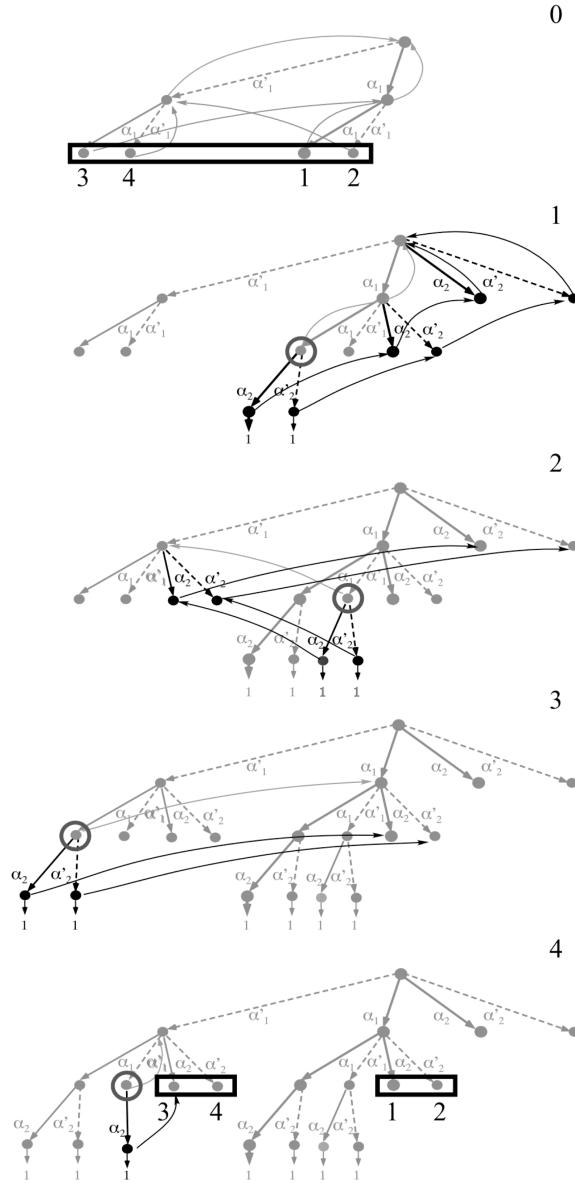


FIG. 6.5 – Insertion du symbole α_2 , troisième symbole de la structure « $\alpha_1\alpha_1\alpha_2\alpha_1\alpha_2\alpha_1$ » dans un arbre dégénéré ($k = 3$, $\delta_{max} = 1$, $\Delta_{max} = 2$ et comme dans la figure 6.2(a) une seule des deux dégénérances possibles est représentée : $\alpha_1 \rightarrow \alpha'_1$).

0 : Arbre avant l'insertion de α_2 , seuls les deux premiers α_1 , α_1 sont présents. Les noeuds de la liste horizontale sont encadrés. Tous les liens supplémentaires sont visibles, mais dans les schémas suivants seuls les nouveaux liens supplémentaires sont dessinés. Chaque schéma (0, 1, 2, 3, 4) présente l'insertion de α_2 et ses dégénérés à partir d'un des noeuds de la liste horizontale.

1 : Insertion à partir du premier noeud de la liste horizontale, celui ayant le motif le moins dégénéré. La liste verticale (les liens supplémentaires sont en gris) est parcourue. Deux feuilles sont d'abord créées. Le premier lien supplémentaire est suivi et deux noeuds et deux liens supplémentaires (pour les nouvelles feuilles) sont créés (les flèches noires ascendantes). Le lien supplémentaire suivant est suivi et deux noeuds sont créés sous la racine, leur supplément est donc la racine, les deux liens supplémentaires pour les noeuds précédents sont aussi créés .

2 : Insertion à partir du second noeud de la liste horizontale. Comme à l'étape précédente, les liens supplémentaires visibles sur le schéma sont suivis et les noeuds ou feuilles et leurs liens supplémentaires sont créés. Il n'est nécessaire de remonter que jusqu'au niveau 1 car les symboles ont déjà été insérés au niveau 0 à l'étape précédente.

3 : Insertion à partir du troisième noeud de la liste horizontale. Ici, il suffit de créer les feuilles et leur lien supplémentaire (niveau 2) : le premier symbole du motif du noeud de la liste est dégénéré.

4 : Insertion à partir du troisième noeud de la liste horizontale. Comme à l'étape précédente, il suffit de créer les feuilles et leurs liens supplémentaires mais pour respecter la condition Δ_{max} , il ne faut pas insérer de symbole dégénéré mais uniquement le symbole exact.

Le prochain symbole sera inséré en utilisant la liste horizontale encadrée, les opérations seront les mêmes.

Algorithm 1 Algorithme de construction de l'automate, motif de longueur k.

Remarques :

Un noeud possède autant de noeuds fils qu'il y a de symboles dans l'alphabet et possède un lien suppléant.

Une feuille n'a pas de noeud fils mais contient toutes les positions du motif formé par les symboles de la branche.

Chaque élément ou point d'insertion (**Point**) de la liste horizontale contient un noeud et la liste des δ associés aux symboles de la branche (ainsi que la somme de ces δ grâce à laquelle on vérifie la condition Δ_{max}).

```

root ← NewNode() ;
UpdateList(List, root, 0, 0) ;
for all Pos de la structure requête do
    for all Point of list do
        for all paires ( $S_i, \delta_i$ ),  $\delta_i \leq \delta_{max}$  and  $|\delta_i|$  augmentant do
            if Point.sum $\delta + \delta_i \leq \Delta_{max}$  then
                //Première étape : Création du nouveau nœud ou feuille au niveau le plus bas
                if Pos < k then
                    Point.node.child[S] ← NewNode() ;
                else
                    if Point.node.child[S] <> NIL then
                        Point.node.child[S] ← Newleaf() ;
                    end if
                    AddPos(Point.node.child[S], Pos) ;
                end if
                //Seconde étape : parcourir la liste verticale en insérant le symbole à chaque niveau et mettre à jour les liens suppléants
                if Point.δList[1] == 0 then
                    InsertGoBackUp( $S_i$ , Point.node.substitute, Point.node.child[S $_i$ ]) ;
                    UpdateList(nextList, Point.node.child[S $_i$ ]).substitute, Point.δList,  $\delta_i$ ) ;
                else
                    Point.node.child[S $_i$ ].substitute ← Point.node.substitute.child[S $_i$ ] ;
                end if
            end if
        end for
    end for
    list ← nextList ;
end for
    
```

//La liste construite est utilisée pour l'insertion du symbole suivant..

NewNode crée un nœud. La fonction retourne le nœud.

Newleaf crée une nouvelle feuille. La fonction retourne la feuille créée.

AddPos(leaf, Pos) ajoute une position à la liste de positions d'une feuille.

UpdateList(horizontal list, node, list of its δ , current δ) met à jour la liste horizontale (ajout du suppléant du nœud) et la liste des δ (enlève le 1er δ et ajoute le δ courant à la fin), calcule la nouvelle somme des δ .

que les transitions dites « **succès** » i.e. associées aux symboles des motifs présents dans S_r (ou similaires). Il manque deux types de transitions : les transitions « **échec** » correspondant aux symboles de motifs non rencontrés lors de la construction et les transitions « **fin** » permettant de poursuivre la recherche lorsqu'un motif a été trouvé, c'est-à-dire lorsqu'on est arrivé à une feuille. Les transitions

Algorithm 2 Fonction InsertGoBackUp, parcourir la liste verticale

InsertGoBackUp(symbol S, node currentNode, node previousNode)

currentNode : nœud en dessous duquel un nouveau nœud est créé.

previousNode : nœud créé précédent, son lien suppléant sera le nouveau nœud.

```

if currentNode.child[S] == NIL then
    currentNode.child[S] ← NewNode() ;
    if currentNode.substitute <> NIL then
        InsertGoBackUp(S, currentNode.substitute, currentNode.child[S]) ;
        //à partir du nœud suppléant du nœud courant (pas son fils le nouveau nœud)
    else
        currentNode.child[S].substitute ← currentNode ;
        // Si on est à la racine, juste mettre à jour le lien suppléant du nouveau nœud.
    end if
end if
previousNode.substitute ← currentNode.child[S] ; //Le suppléant du nœud créé précédent (au niveau inférieur) est le nouveau nœud.
    
```

//Si le symbole n'est pas déjà présent,
//créer le nœud.
//Si on n'est pas à la racine,
//répéter l'opération au niveau supérieur.
// Si on est à la racine, juste mettre à jour le lien suppléant du nouveau nœud.
//le suppléant est la racine.

succès sont toujours descendantes tandis que les transitions fin et échec sont soit horizontales soit ascendantes (au pire, elles remontent jusqu'à la racine).

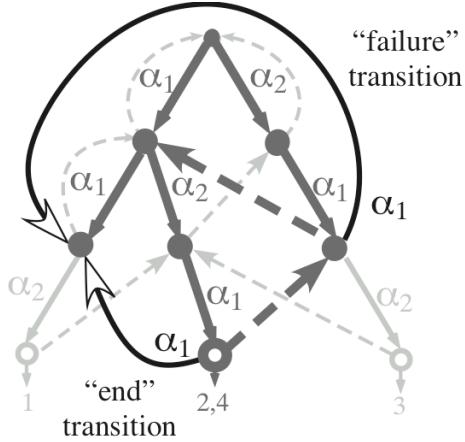


FIG. 6.6 – Automate (incomplet) non dégénéré avec tous les liens suppléants. La structure et les paramètres sont toujours les mêmes. Les flèches pointillées ascendantes sont les liens suppléants. Les deux autres flèches ascendantes représentent les deux types de transitions : transition fin et transition échec (toutes les transitions échec et fin ne sont pas représentées). Les transitions échec sont définies pour éviter de comparer plusieurs fois un symbole du texte lorsqu'il n'est pas présent dans les transitions succès.

Par exemple, si le texte est $\alpha_2\alpha_1\alpha_1$, les deux premiers symboles sont trouvés en partant de la racine mais le troisième symbole (α_1) n'est pas présent en dessous du nœud de motif $\alpha_2\alpha_1$. Il faut alors remonter au suppléant de ce nœud (de motif α_1) et activer la transition α_1 de ce nœud. Pour éviter ces deux étapes, une transition échec (*failure transition*) est définie, partant du nœud de motif $\alpha_2\alpha_1$, arrivant au nœud de motif $\alpha_1\alpha_1$ et associée au symbole α_1 (le sous-motif $\alpha_1\alpha_1$ est bien celui que l'on vient de lire dans le texte).

La transition fin (*end*) permet de même de ne suivre qu'une transition au lieu de deux liens suppléants (en flèches ascendantes gris foncé) et une transition succès lorsque le texte est $\alpha_1\alpha_2\alpha_1\alpha_1$.

L'automate incomplet construit permet, lorsqu'un motif est trouvé, de poursuivre la recherche en allant au nœud suppléant N' de la feuille N . La transition de ce nœud suppléant correspondant au symbole suivant dans le texte est alors activée. Si aucune transition de ce nœud N' ne correspond au symbole, il suffit d'aller à son nœud suppléant N'' et de continuer. Cependant, pour éviter de comparer plusieurs fois ce symbole et de remonter autant de fois qu'on ne le trouve pas il est utile d'avoir une transition menant directement au nœud N_f auquel on serait arrivé par remontées successives. Ce nœud N_f a donc pour motif le plus long suffixe de N présent dans l'arbre (voir figure 6.6 pour des exemples). Le motif de N_f est le préfixe d'un motif de longueur k . Toute l'opération revient en fait à chercher les préfixes qui sont les plus longs suffixes de tous les motifs de longueur 2 à k .

Ces deux types de transition sont définis en parcourant l'arbre en entier de la racine jusqu'aux feuilles et en largeur d'abord. Ainsi, les transitions manquantes sont définies niveau par niveau. Au niveau 1 les transitions échec pointeront toutes vers la racine (elles sont associées à des symboles qui n'ont jamais été rencontrés lors de la construction). Aux niveaux inférieurs, si un symbole n'a pas de transition en dessous d'un nœud, il suffit de recopier la transition définie au nœud suppléant pour ce symbole.

Finalement, il y a une transition pour chaque symbole de l'alphabet à chaque nœud, et chaque symbole du texte n'active qu'une seule transition (voir figure 6.2(b) pour l'automate final).

6.1.4 Recherche, sélection et extension des graines

Les motifs communs (ou **graines**) aux structures de la banque et à la structure requête sont recherchés en utilisant l'automate. Les positions des motifs trouvés sont conservées. Généralement, il y a plusieurs milliers de graines pour une paire de structures mais seules les meilleures de ces graines seront étendues et deviendront des SHSP.

Il est possible de représenter les graines trouvées dans un graphique avec en abscisse et en ordonnée les indices des résidus dans les deux structures (voir figure 6.7).

Dans un premier temps, les graines qui sont sur la même diagonale et proches sont rassemblées pour former des graines plus longues. Ce filtre permet d'éliminer les motifs répétitifs dus aux structures secondaires répétées. Ces motifs répétitifs sont visibles dans le graphique 6.7 sous forme de rectangles plus ou moins rognés aux coins. Seuls les plus longs de ces motifs sont conservés. Le filtre consiste à ne conserver que les graines qui ne sont incluses dans aucune autre (mais elles peuvent être chevauchantes). Seules quelques dizaines de graines satisfont ces deux conditions.

Les graines subsistantes sont ensuite étendues pour former les SHSP. L'algorithme en O(N) utilisé (Gries, 1982) permet de trouver les segments de score maximaux, *i.e.* les séries de résidus dont la somme des score est maximale. Ce score est basé sur la différence entre les angles α réels - et non avec les symboles entiers utilisés jusqu'à présents. Nous utilisons en plus une heuristique qui permet d'arrêter l'extension si le score chute en dessous d'un certain seuil, les différences angulaires étant alors trop grandes pour avoir une chance d'être compensées. Le score est calculé selon l'équation :

$$S_{sim} = \sum_{i=0}^l (T - D_a(\alpha_{q+i}, \alpha_{b+i})) \quad (6.1)$$

où $D_a(\alpha_{q+i}, \alpha_{b+i})$ est la différence angulaire entre deux angles α_{q+i} et α_{b+i} , l la taille de la graine, T une constante, α_q l'angle d'indice q de la structure requête et α_b l'angle d'indice b de la structure de la banque.

Plus deux angles sont proches plus le score est grand. Nous avons calculé la différence moyenne de deux angles pris au hasard dans la PDB, elle est d'environ 87° . Environ un quart des différences d'angles pris au hasard sont inférieures à 30° . Il faut que le score entre deux angles pris au hasard soit en moyenne négatif dans l'algorithme utilisé pour que le score ne soit pas toujours croissant le long d'une extension, T a donc pour valeur 30° . La méthode d'extension est expliquée dans la figure 6.8. Il faut ajouter une autre contrainte pour que les SHSP soient similaires en structure : la différence entre deux angles doit être inférieure à un seuil (généralement on prend 60°). Enfin, les graines étendues ne sont conservées et ne deviennent des SHSP que si elles sont suffisamment longues.

²⁰Ces paramètres ne sont pas les plus couramment utilisés mais il y a ainsi moins de graines donc le graphique est plus lisible.

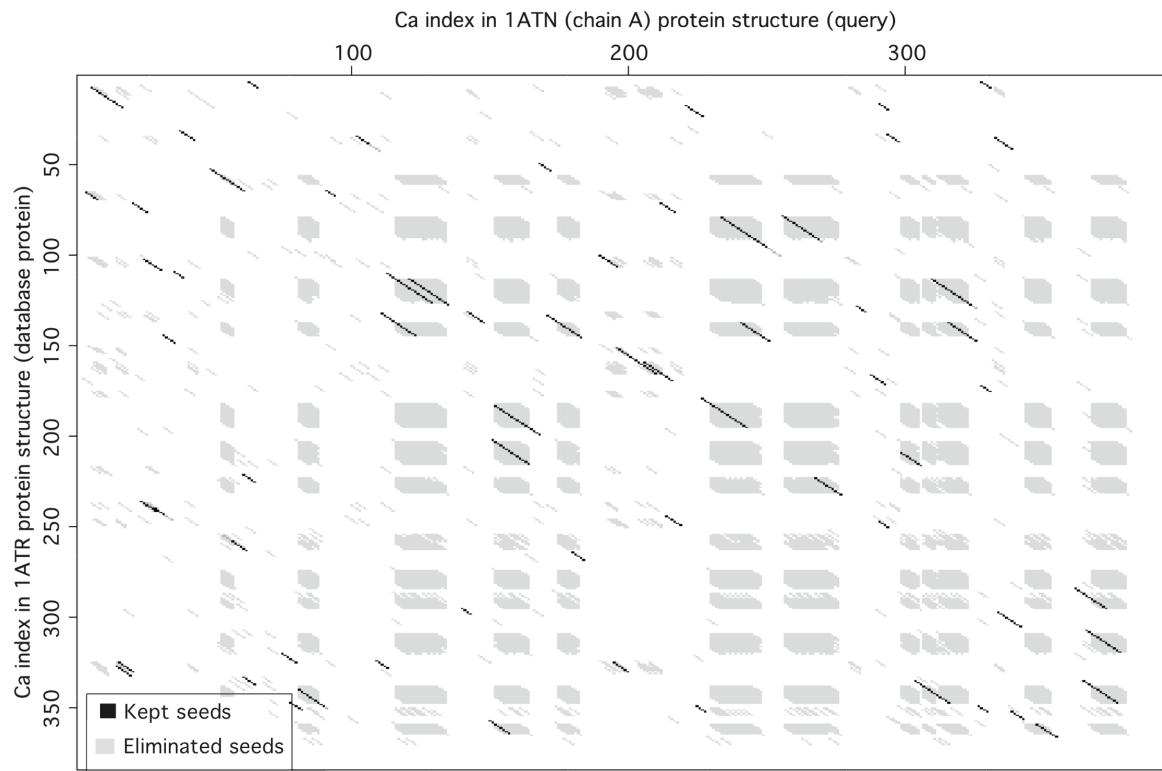


FIG. 6.7 – Graines trouvées lors de la comparaison des structures 1atn (une endodéoxyribonucléase de *Oryctolagus cuniculus*) et 1atr (une protéine de choc thermique de *Bos taurus*). Les motifs de l’automate (k) étaient de longueur 6, la dégénérescence locale δ_{max} de 1 et la dégénérescence globale Δ_{max} de 7^{20} . Les rectangles rognés aux coins sont des concentrations de graines qui correspondent à des structures canoniques répétées (généralement des hélices α). Les graines en noir sont celles conservées après filtrage.

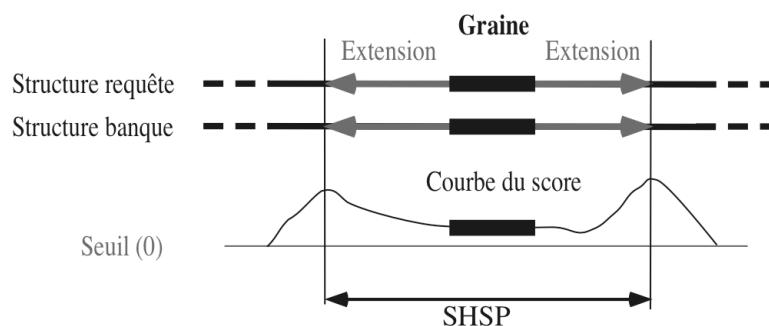


FIG. 6.8 – Extension des graines : les scores angulaires des paires d’angles voisins à droite et à gauche de la graine sont ajoutés au score de celle-ci. Tant que le score total est au-dessus du seuil, l’extension se poursuit, sinon elle s’arrête. Les bornes du SHSP sont les résidus où le score total est le plus haut.

6.1.5 Sélection des SHSP

Sont considérés comme incompatibles, les SHSP qui se « chevauchent » dans l'une des deux protéines ou qui n'ont pas le même ordre séquentiel dans les deux protéines (voir figure 6.9). La seconde condition aurait pu être omise - sans compliquer l'algorithme - pour pouvoir trouver les permutations circulaires de structures ou les enchaînements différents de sous-structures similaires. Cependant, il semble ces cas soient assez rares et que le nombre de résultats erronés engendrés aurait été trop important.

Il y a en général quelques dizaines de SHSP au total pour chaque paire de structures. Il est donc possible de regarder tous les « **chemins** » possibles parmi les SHSP et de choisir celui de meilleur score. Ce score global ne dépend que des scores des SHSP composant le chemin ; aucune pénalité n'est accordée aux gaps.

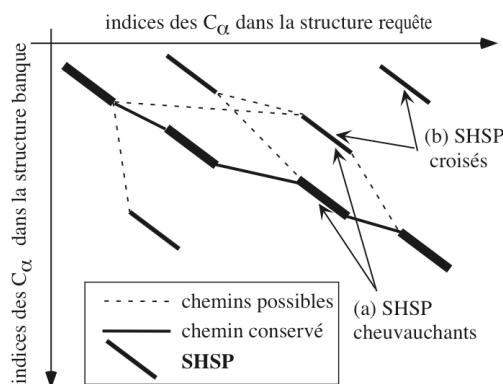


FIG. 6.9 – Sélection des SHSP compatibles : ils ne peuvent être ni croisés (a) ni chevauchants (b). Tous les chemins possibles sont représentés par les pointillés, mais seuls les SHSP du meilleur chemin sont conservés.

6.1.6 Calcul du score final et classement

La plupart des structures de la PDB ont au moins une sous-structure commune avec une structure requête, même si elle est non significative (voir plus loin). Il est donc nécessaire de calculer un score permettant de classer les structures de la banque selon les similarités structurales trouvées. Plusieurs scores ont été testés.

Le premier score (S_{sim}), calculé lors de l'étape d'extension de chaque SHSP du chemin, est la somme des scores S_a (c.f. équation 6.1). En utilisant ce score, les premières structures classées sont les structures très similaires à la structure requête. Le classement des autres protéines est moins adéquat : des structures similaires mais éloignées de la structure requête peuvent être classées après des structures pourtant différentes de la structure requête. En effet, si des hélices α sont dans les SHSP, le score S_{sim} est très fort car les angles α dans les hélices varient peu et les hélices sont souvent longues par rapport à la taille d'un SHSP. Ainsi, les structures ayant beaucoup d'hélices α sont très souvent bien classées même si elles n'ont pas d'autre ressemblance avec la structure requête. Pour éviter ce

phénomène, le second type de score est fondé sur la probabilité de trouver un SHSP (plus précisément la probabilité de trouver le fragment structural du SHSP provenant de la structure de la banque). Cette probabilité pourrait être calculée comme suit :

$$Score = \sum_{s=0}^{NS} \sum_{i=0}^{n_s} \log(P(\alpha_{s_i})) \quad (6.2)$$

où NS l'est le nombre de SHSP de la paire de structures, n_s le nombre de résidus dans le s^e SHSP, $P(a)$ la probabilité de l'angle de valeur a dans la PDB, α_{s_i} est la valeur du i^e angle α dans le s^e SHSP.

Cependant, dans les structures protéiques, les angles α sont loin d'être indépendants. Un modèle de Markov d'ordre 1 de ces dépendances pourrait être calculé, mais un ordre supérieur - nécessaire ici - demanderait l'estimation de plus de paramètres qu'il n'est possible d'en calculer avec les données actuelles. Nous avons donc calculé une approximation par paires d'un modèle de Markov, un modèle MTD (*Mixture Transition Distribution Model* (Raftery, 1985; Berchold and Raftery, 1999)).

Un modèle MTD est une approximation d'un modèle de Markov par les probabilités conditionnelles des paires. Soit X_t une variable aléatoire dans un espace fini $A = \{1, \dots, m\}$. Dans un modèle de Markov d'ordre ℓ la probabilité que $X_t = \alpha_0$ ($\alpha_0 \in A$) dépend de la combinaison des valeurs prises par $X_{t-\ell}, \dots, X_{t-1}$. Dans un modèle MTD, la probabilité dépend des probabilités des paires X_t, X_{t-i} chacune multipliée par un facteur de « retard ». Elle se calcule selon la formule :

$$P(X_t = \alpha_0 | X_{t-\ell} = \alpha_\ell, \dots, X_{t-1} = \alpha_1) = \sum_{g=1}^{\ell} \lambda_g P(X_t = \alpha_0 | X_{t-g} = \alpha_g) = \sum_{g=1}^{\ell} \lambda_g q_{\alpha_g \alpha_0} \quad (6.3)$$

où $\alpha_\ell, \dots, \alpha_0 \in A$, $\lambda = (\lambda_\ell, \dots, \lambda_1)'$ est un vecteur de paramètres de « retard » et les probabilités $q_{\alpha_g \alpha_0}$ sont des éléments d'une matrice de transition $Q = [q_{ij}]$ de taille $m \times m$. Pour que les résultats du modèle soient bien des probabilités, λ doit respecter les contraintes :

$$\sum_{g=1}^{\ell} \lambda_g = 1 ; \lambda_g \geq 0 \quad (6.4)$$

Ce modèle a donc $m(m-1) + \ell - 1$ paramètres indépendants, tandis qu'un modèle de Markov en aurait $m^\ell - 1$.

Dans notre cas, l'espace de la variable X_t est l'espace des angles α discrétisés selon une maille de 10° , c'est donc une matrice 36×36 . Le modèle a été estimé avec une banque de structures non-redondantes extraites de la PDB et $\ell = 8$. Le score S_{MTD} du meilleur chemin de SHSP de chaque paire de structures est le produit des probabilités de chaque SHSP calculée selon ce modèle (on l'exprime plutôt en somme des logarithmes des probabilités).

Avec ce score, les paires de structures ayant des similarités dues uniquement aux structures secondaires sont alors moins bien classées, mais certaines similarités structurales réelles étaient toujours mal classées. Nous avons alors utilisé un nouveau score S_{comp} qui est un score plus global car seuls les SHSP **compatibles spatialement** (voir figure 6.10) sont considérés. Pour évaluer cette comptabilité

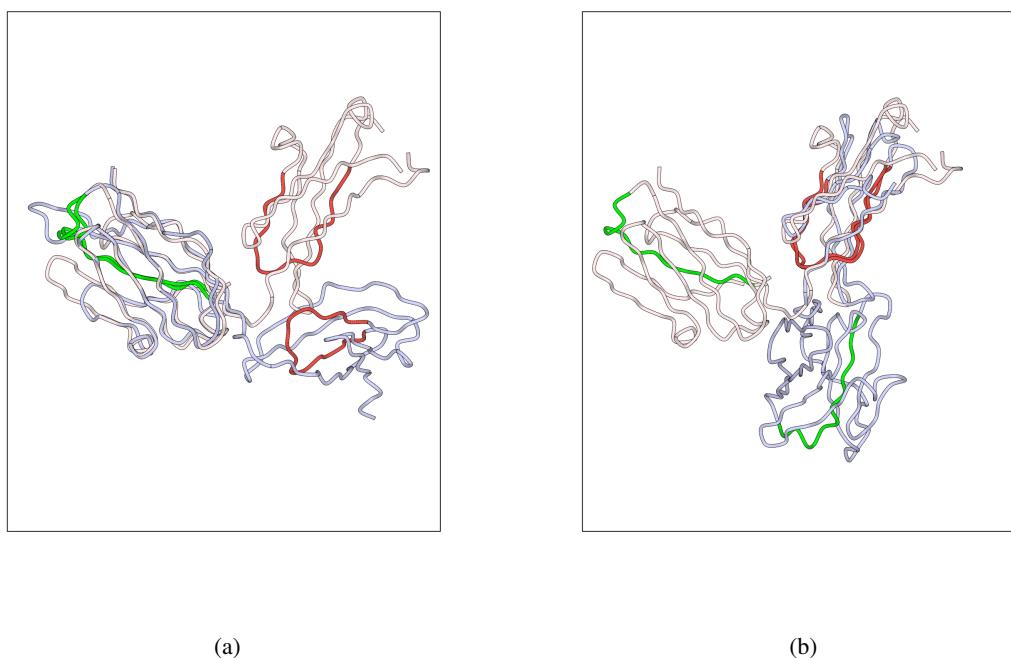


FIG. 6.10 – Alignement par YAKUSA de deux structures d’Immunoglobulines : 1MEX chaîne H en rouge, et 1HXM chaîne A en bleu. Les blocs structuraux similaires sont de couleurs différentes pour chaque bloc. Dans la figure 6.10(a), les deux structures sont alignées selon le bloc vert du domaine gauche ; dans la figure 6.10(b) les deux structures sont alignées selon le bloc rouge du domaine droit. Il est visible que les deux domaines bien qu’ils soient similaires ne peuvent pas être superposés en même temps.

spatiale le $RMSD_c$ utilisé est tel que pour chaque SHSP d'une paire de structure :

- la superposition de la structure requête et de la structure de la banque est effectuée selon ce SHSP uniquement ;
- le RMS de tous les autres SHSP est calculé dans cette conformation des deux structures (ces $RMSD_c$ résultants sont nommés $RMSD_c$ croisés).

Deux SHSP sont considérés comme compatibles spatialement si leurs deux $RMSD_c$ croisés sont inférieurs à un seuil (généralement 15 Å). Cela indique que les deux sous-structures ont grossièrement la même orientation dans les deux protéines. Comme nous l'avons vu, il n'est pas rare que deux structures partagent plusieurs structures secondaires mais il est improbable que celles-ci aient les mêmes orientations si les deux structures ne sont pas similaires (sauf si les SSE sont dans deux domaines différents ayant « bougé »). Ainsi, le dernier score S_{comp} est un score S_{MTD} calculé uniquement avec les SHSP compatibles spatialement. C'est donc un score plus global que le S_{MTD} .

Comme la banque entière de structures est effectivement parcourue pour chaque structure requête, il est possible de calculer un Z-score :

$$\text{Z-score} = \frac{S_{comp} - \bar{S}_{comp}}{\sigma_{S_{comp}}} \quad (6.5)$$

6.1.7 Complexité - Temps d'exécution

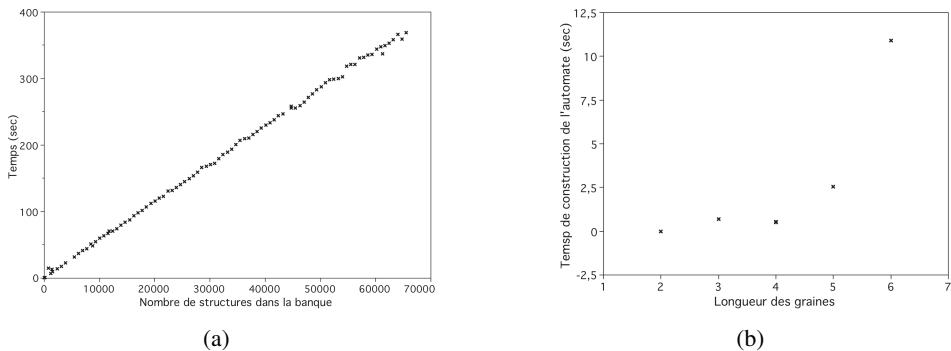


FIG. 6.11 – (a) Temps d'exécution de YAKUSA en fonction du nombre de structures dans la banque.
(b) Temps de construction de l'automate en fonction de la longueur des graines

La recherche croît linéairement avec la taille de la banque. Pour la construction de l'automate, si N est la longueur de la requête, k la longueur des graines, et D la dégénérescence locale maximale (*i.e.* le nombre de symboles voisins considérés comme similaires), la complexité dans le pire des cas est en $N * k * D^k$. En effet, on doit créer au maximum $N - k + 1$ motifs de longueur k , chacun donnant D^k motifs voisins (y compris lui-même) et pour chacun de ces motifs, insérer k symboles. En réalité, les répétitions internes des motifs et les répétitions de motifs similaires sur la structure font que beaucoup de motifs sont déjà présents ou partiellement présents, et on a vu que l'algorithme de construction ne

« repasse » pas par les motifs ou parties de motifs déjà créés. De plus, la dégénérescence globale limite le nombre de motifs voisins générés par la dégénérescence locale. La complexité réelle de ce problème est donc plus faible (des tests nous ont montré que le temps de construction de l'automate n'est pas réellement exponentiel avec la dégénérescence). En fait, le temps de construction de l'automate est négligeable devant le temps de recherche sur la banque (le rapport est de l'ordre d'un deux centièmes avec les banques actuelles, voir figure 6.11). Sur un Macintosh G5 1.2 Ghz, le temps de construction de l'automate est de l'ordre de 0.4 secondes, et la recherche totale sur une banque de 8000 structures prend environ 45 secondes.

6.1.8 Implémentation

YAKUSA a été implémenté en langage C et pour les systèmes Unix (Macintosh OS X, Linux OS et IRIX OS).

Plusieurs banques de structures ont été prétraitées (calcul des angles α) et sont en format binaire pour plus la rapidité de lecture :

- une banque non-redondante où les structures n'ont qu'une chaîne, moins de 80% d'identité de séquence protéique et plus de 50 résidus,
- les banques ASTRAL (Brenner et al., 2000) ;
- les banques Pisces (Wang and R. L. Dunbrack, 2003) ;
- la PDB entière ;
- le sous-ensemble de la PDB « cluster50 » des structures dont les séquences présentent moins de 50% d'identité entre elles ;
- le sous-ensemble de la PDB « cluster70 » des structures dont les séquences présentent moins de 70% d'identité entre elles.

Beaucoup de paramètres de YAKUSA sont réglables. Leurs valeurs par défaut sont :

- maille pour la discréétisation des angles α : 10° (voir section « Description de la structure tridimensionnelle des protéines ») ;
- longueur des graines k : 4 résidus (voir section « Description générale de l'algorithme » et suite) ; ce paramètre joue sur la finesse de la recherche (et sur le temps !) ;
- dégénérescence locale maximale δ_{max} : 3 en unité de maille (voir paragraphe « La dégénérescence ») ; ce paramètre joue sur le flou des graines trouvées ;
- dégénérescence locale maximale Δ_{max} : 7 en unité de maille (voir paragraphe « La dégénérescence ») ;
- longueur minimale des SHSP : 6 résidus (voir section « Recherche, sélection et extension des graines ») ;
- nombre maximal de résidus entre deux graines pour qu'elles soient combinées : 10 résidus (voir section « Recherche, sélection et extension des graines ») ;
- constante T du score S_{sim} : 30° (voir section « Recherche, sélection et extension des graines ») ; ce paramètre joue sur l'extension des SHSP ;

- différence maximale autorisée entre deux angles α : 60° (voir section « Recherche, sélection et extension des graines »).

Avec ces paramètres, le nombre de nœuds de l'automate est généralement de l'ordre du million. Les trois paramètres k , δ_{max} et Δ_{max} influencent grandement les SHSP trouvés ; plus k est petit et δ_{max} et Δ_{max} sont grands, plus il est possible de trouver des similitudes lointaines entre deux structures. Néanmoins les paramètres par défaut semblent être les plus efficaces (les résultats ne changent presque pas en diminuant k et augmentant δ_{max} et Δ_{max}). Les SHSP de taille inférieure à 6 résidus correspondent le plus souvent à des ressemblances structurales dues au hasard même dans les structures similaires (par exemple le SHSP peut être composé de deux petits brins β localisés à des endroits totalement différents dans les deux protéines). Le nombre maximal de résidus entre deux graines n'a pas beaucoup d'influence tant que les valeurs restent raisonnables (inférieures à 20). Les valeurs des deux derniers paramètres sont expliquées dans « Recherche, sélection et extension des graines » page 108.

6.2 Tests et résultats

YAKUSA a été comparé à 12 autres méthodes de comparaison de structures (table 6.2) pour des cas variés. Plusieurs tests ont été effectués : i) un test sur les serveurs, ii) un test de sensibilité aux petites variations de structures en utilisant des modèles RMN, iii) un test pour les protéines multi-domaines et iv) un test pour des structures requêtes possédant un nouveau repliement. Ces tests sont fondés sur une évaluation.(Novotny et al., 2004) récente de serveurs de comparaison de structures et sur une comparaison entre 3D-Hit et DALI faite par J. Gough²¹ avec la banque LiveBench 4 (Fischer and Rychlewski, 2003). En plus, nous avons réalisé une autre évaluation avec des protéines « difficiles » sélectionnées par Fisher et coll. (Fischer et al., 1996).

6.2.1 Test général

Ce premier test est basé sur la classification de CATH : dix familles de protéines ont été sélectionnées comme représentatives des 4 classes principales de CATH (voir page 89). Dans ce test, il y a de 4 à 9 protéines par classes. Le but est de trouver au moins l'une des protéines de la famille avec chacune des structures prises tour à tour comme requête. Les résultats pour les autres programmes sont de Novotny et coll. (Novotny et al., 2004) et nous avons exécuté YAKUSA sur les mêmes structures. Comme chaque programme possède sa base de données, nous avons utilisé la banque non redondante usuelle de YAKUSA. Pour chaque structure dans cette base, nous avons une liste des protéines qui lui sont liées mais qui ne sont pas dans la base puisqu'elles ont 80% d'identité de séquence avec elle (« protéines équivalentes »). Nous considérons que si une protéine est sélectionnée lors d'une requête sur la base, toutes les « protéines équivalentes » qui lui sont liées sont aussi sélectionnées.

²¹<http://supfam.mrc-lmb.cam.ac.uk/LiveBench/>

Programme	url	références
3D-Hit	http://bioinfo.pl/3D-Hit	(Plewczynski et al., 2002)
CE	http://cl.sdsc.edu/ce.html	(Shindyalov and Bourne, 1998)
DALI	http://www.ebi.ac.uk/dali	(Holm and Sander, 1993)
DEJAVU	http://xray.bmc.uu.se/usf/dejavu.html	(Holm and Sander, 1996a)
LOCK	http://gene.stanford.edu/LOCK	(Kleywegt and Jones, 1997)
MATRAS	http://biunit.aist-nara.ac.jp/Matras	(Madsen and Kleywegt, 2002)
PRIDE	http://hydra.icgeb.trieste.it/pride	(Singh and Brutlag, 1997)
SSM	http://www.ebi.ac.uk/msd-srv/ssm	(Kawabata, 2003)
TOP	http://bioinfo1.mbfys.lu.se/TOP	(Carugo and Pongor, 2002)
TOPS	http://balabio.dcs.gla.ac.uk/tops	(Krissinel and Henrick, 2003)
TOPSCAN	http://www.bioinf.org.uk/topscan	(Lu, 2000)
VAST	http://www.ncbi.nlm.nih.gov/Structure/VAST/	(Gilbert et al., 1999)
YAKUSA	http://www.rpbs.jussieu.fr/Yakusa	(Martin, 2000)
		(Gibrat et al., 1996)
		(Carpentier et al., 2005)

TAB. 6.2 – Méthodes de comparaison de structures protéiques testées .

Les résultats sont présentés dans le tableau 6.3. Les trois programmes qui se comportent globalement le mieux sont YAKUSA, CE and DALI. Un point intéressant est que ces trois programmes utilisent des coordonnées internes (distances internes pour DALI et CE, angles α pour YAKUSA) , alors que les autres (à part PRIDE) utilisent les SSEs - au moins comme points de départ - pour les alignements structuraux.

Pour ce test, YAKUSA montre les meilleurs résultats globaux (95% de succès) et échoue seulement dans 3 cas : 2 dans la classe *Mainly α* et 1 dans la classe *mixed α – β* . De plus aucun autre programme ne donne de meilleurs résultats pour la classe *Mainly α* (17/19) bien que nous aurions pu penser que les hélices α pouvaient être un point faible de YAKUSA. Pour les classes *mainly β* et *Few SSEs*, YAKUSA trouve toutes les cibles. Pour la classe *mixed α – β* , seul VAST trouve toutes les cibles. Il peut y avoir des explications pour les 3 échecs de YAKUSA. YAKUSA, comme DALI et CE, échoue à trouver des protéines de la famille 1rlr, une ribonucléotide reductase appartenant à la famille CATH 1.10.40, mais cette protéine n'est pas classée dans la même classe de SCOP que les autres protéines sélectionnées (la classe de SCOP pour 1rlr est a.98.1 alors que celle des 6 autres est a.127.1. Le domaine de 1rlr considéré par CATH est un bouquet de 4 hélices, les autres membres de la classe 1.10.40 de CATH sont aussi des bouquets de 4 hélices, mais avec des hélices nettement plus courtes. YAKUSA, comme DALI et CE, échoue aussi pour la protéine 1c3u (chaîne A) sélectionnée dans le test comme un membre de la famille des L-2-haloacid dehalogénases (1.10.164), mais dans

la classification CATH, les 3 domaines de 1c3u sont classés dans les familles 1.10.275, 1.20.200 et 1.10.40. Le troisième échec est pour la structure 1cfr de la famille des endonucléases de restriction. L'inspection visuel de l'alignement structural de 1cfr avec 1fok (qui appartient à la famille) donné par CE ou DALI (qui ne s'accordent pas pour la partie commune de l'alignement) ne montre qu'une superposition assez grossière des 2 hélices α et des 2 feuillets β qui sont assez différents bien qu'à peu près dans la même orientation. En ce qui concerne les SSE, VAST devrait trouver cette similarité mais même dans la base de VAST, 1cfr n'est pas considéré comme un voisin structural de 1fok (pas plus que les autres membres de la famille).

Le tableau 6.4 montre les résultats détaillés pour la famille des cyclophilines pour tous les serveurs testés. YAKUSA comme CE atteint toutes les cibles pour cette famille : la structure requête - ou une structure équivalente - est toujours trouvée et au moins une protéine de la famille est toujours trouvée. En réalité, tous les membres sont toujours tous trouvés par YAKUSA avec un rang inférieur à 12. Le temps pour une requête de YAKUSA est parmi les plus courtes. PRIDE et TOPSCAN ont aussi des temps d'exécution court, mais leurs résultats sont moins bons (respectivement 62% et 70% de succès, voir tableau 6.3). Evidemment, les temps dépendent du serveur et de la taille de la base de donnée et ne reflètent pas complètement la rapidité du programme. DALI par exemple peut durer quelques heures quand la requête est un repliement unique et n'a pas d'homologue dans le *representative set* de DALI. De plus, dans ce cas, DALI utilise un pré-filtre basé sur les SSE et n'examine donc pas tous les repliements de la banque. Ceci pourrait expliquer la meilleure sensibilité de CE par rapport à DALI (Novotny et al., 2004). Comme CE utilise des résultats précompilés, les nouveaux repliements sont comparés à la base au niveau des séquences, et si aucune similarité de séquence n'est détectée, CE peut durer aussi quelques heures puisqu'il a alors à traiter une banque locale de structures représentatives. En contraste, pour chaque requête, YAKUSA traite les 11750 structures de sa base sans préfiltre en moins d'une minute.

6.2.2 LiveBench

LiveBench (Fischer and Rychlewski, 2003; Rychlewski et al., 2003) est un métaserveur utilisé pour classer l'efficacité des programmes de prédiction de structure. A une date donnée, la base de LiveBench contient les nouvelles structures de la PDB qui ne sont similaires en séquences à aucune autre structure connue de la PDB (*E-value cutoff* de 0.001 dans BLAST). LiveBench n'est pas vraiment un *benchmark* pour les programmes de recherche de similarités structurales. Mais avec la version 4 de la base Livebench (106 structures), une évaluation de tous les programmes de prédiction a été faite par J. Gough, qui les a comparé aux résultats de DALI et de 3D-HIT(Plewczynski et al., 2002). 3D-Hit⁽²²⁾ est aussi un programme de comparaison de structures fondé sur un algorithme de recherche rapide basé sur du hachage de distances internes détectant les similarités entre des blocs d'au maximum 300 résidus (voir page 35). L'évaluation de J. Gough utilise la classification structurale de SCOP(Conte et al., 2002). Dans cette évaluation des programmes de prédiction, si la structure cible et la structure

²²url : <http://bioinfo.pl/3D-Hit>

Programme	<i>Mainly α</i>	<i>Mainly β</i>	<i>Mixed α – β</i>	<i>Few SSEs</i>	total
nombres de structures	19	19	15	8	%
YAKUSA	17	19	14	8	95%
CE	17	19	13	8	93%
DALI	14	19	14	8	90%
MATRAS	11	19	14	8	85%
VAST	12	17	15	7	84%
TOP	14	18	12	7	84%
DEJAVU	14	19	9	4	75%
TOPSCAN	15	12	9	7	70%
TOPS	2	15	14	7	62%
PRIDE	14	14	7	3	62%
LOCK	0	14	11	8	54%
SSM	5	13	10	5	54%

TAB. 6.3 – Performance des programmes selon les structures représentatives des quatre classes principales de CATH. Par exemple, 19 structures de la classe *Mainly α* ont été soumises aux serveurs, et pour 17 d'entre elles, YAKUSA et CE ont trouvé au moins une autre structure de la même famille parmi les 100 premiers résultats. Toutes les données (sauf pour YAKUSA) sont tirées de l'article de Novotny et coll.(Novotny et al., 2004).

Programme	Trouve la requête	Autres vrais positifs	Rang du premier faux positif	Durée de la requête (min)
YAKUSA	9	9	14	<1
CE	9	9	rien trouvé ou rang 4	(utilise des résultats précompilés)
DALI	2	9	rien trouvé ou rang 4	3-9
DEJAVU	9	9	rien trouvé	5-55
LOCK	1	8	2	8-15
MATRAS	3	9	3	10-45
PRIDE	6	8	1-8	<1
SSM	6	8	rien trouvé	2-3
TOP	7	9	rien trouvé ou rang 6-8	7-10
TOPS	5	9	2-6	2-5
TOPSCAN	5	9	4-6	<1
VAST	2	9	rien trouvé ou rang 3	5-25

TAB. 6.4 – Résultats pour la famille des cyclophilines pour le test général. Toutes les données (sauf pour YAKUSA) sont de l'article de Novotny et coll.(Novotny et al., 2004). Il y a 9 protéines dans cette famille. Par exemple, YAKUSA trouve la requête elle-même pour 9 des 9 protéines (colonne 2) et il trouve au moins une protéine de la famille pour chacune des 9 protéines comme requête (colonne 3)(en fait, pour chaque structure requête in fact, YAKUSA trouve les 8 autres protéines de la famille). Pour YAKUSA, le rang le plus élevé pour un faux positif est 14 (colonne 4), et la recherche dure moins qu'une minute (colonne 5).

« patron » appartiennent à la même super-famille de SCOP, le modèle est jugé comme vrai positif. Puisque l'évaluation contenait les résultats de DALI et de 3D-HIT, elle peut être « détournée » pour être aussi un *benchmark* pour comparer d'autres programmes de recherche de similarités structurales. Nous avons donc exécuté YAKUSA avec chacune des 106 structures de LiveBench 4 comme requête, en utilisant comme base de donnée structurale toutes les structures PDB de la version 1.61 de SCOP 1.61, *i.e.* la même version de SCOP que celle utilisée par J. Cough (table 6.5). Des 106 cibles de l'évaluation, 79 avaient des structures « patrons » existantes, 33 étaient des nouveaux repliements et donc, dans la base utilisée ici, il ne peut y avoir de vrais positifs pour ces 33 repliements.

Programme	YAKUSA	DALI	3D-HIT
1	61	56	51
2	65	57	53
3	67	65	53
4	68	67	55
5	68	67	55
6	68	67	56
7	69	67	56
8	69	67	58
9	69	67	61
10	69	68	61
Sensibilité	69	69	68
Spécificité	67.3	64.8	55.9

TAB. 6.5 – Résultats avec les 106 structures de LiveBench4 (Fischer and Rychlewski, 2003). Chaque ligne de 1 à 10 indique le nombre de vrais positifs (voir texte) avant le n° faux positif, où n est le numéro de ligne. Le score de sensibilité est le nombre total de vrais positifs. Le score de spécificité est la moyenne des 10 lignes, et indique la qualité du programme à bien classer les vrais positifs.

Deux paramètres peuvent être évalués pour une méthode : sa sensibilité et sa spécificité. La sensibilité est la capacité à trouver le maximum de vrais positifs dans toute la liste de structures similaires sélectionnées. La spécificité mesure l'efficacité du programme à classer un vrai positif mieux qu'un faux positif (tableau 6.5).

YAKUSA et DALI trouvent le même nombre de vrais positifs (*i.e.* sensibilité dans le tableau 6.5) et ont de meilleures performances que 3D-HIT. YAKUSA est plus spécifique que DALI et 3D-HIT, *i.e.* il classe mieux les vrais positifs. Par exemple, YAKUSA classe 61 vrais positifs comme premiers, alors que DALI classe seulement 56 vrais positifs comme premiers, et 3D-HIT descend à 51. En égards à ce test, YAKUSA semble donc plus spécifique que DALI ou 3D-HIT.

Ces deux tests montrent les capacités de YAKUSA en conditions réelles. Il a les résultats les meilleurs en un temps le plus court. Il est intéressant de noter que - contrairement à d'autres programmes - les résultats de YAKUSA ne semblent pas dépendants de la composition en SSE des structures.

6.2.3 Sensibilité aux petites variations de structures

Puisque la méthode est locale, il est important de vérifier qu'elle n'est pas trop sensible aux petites variations de structures. Dans l'étude de Novotny, cinq modèles d'une même structure RMN (PDB 2gda, modèles 2, 7, 12, 18 et 20) ont été donnés comme requêtes et les résultats ont été comparés au résultats obtenus pour le modèle moyen minimisé de la même protéine (PDB 1gdc, récepteur au glucocorticoïde, domaine de liaison à l'ADN). Les modèles 2 et 20 ont été choisi parce qu'ils ont respectivement les plus bas et plus haut $RMSD_c$ avec la structure 1gdc.

Dans l'étude de Novotny, CE, LOCK et TOP obtiennent les mêmes résultats pour tous les modèles, alors que les autres programmes sont plus ou moins sensibles aux petites variations. DALI, PRIDE, MATRAS et VAST ont toujours un vrai positif comme premier résultat, mais retrouvent un nombre différent de vrais positifs et de faux positifs pour les différents modèles. Ainsi, ces 4 programmes ne sont pas très affectés par les petites variations structurales. TOPS, TOSCAN et SSM échouent à trouver des structures similaires pour un modèle RMN ou plus. DEJA VU échoue dans tous les cas sauf pour le modèle RMN n°2 de 2gda (avec seulement 1 vrai positif dans la liste). Ces 4 derniers programmes semblent très sensibles aux variations structurales.

YAKUSA trouve 12 vrais positifs avec 1gdc comme requête, mais il y a un faux positif au rang 12. Toutes les 12 structures similaires sont aussi trouvées avec chacun des modèles de 2gda comme requête. Elles sont toutes classées dans les 12 premières sauf pour le modèle 12 où une autre protéine est classée 12^e. Cependant, cette structure semble aussi être un vrai positif - même si elle n'est pas retrouvée avec 1gdc - parce qu'elle est aussi trouvée avec les modèles 2 et 20. Ainsi, YAKUSA semble très légèrement sensible aux petites variations de structures, mais moins que DALI, MATRAS, VAST et PRIDE. Puisque YAKUSA s'appuie sur les alignements structuraux locaux, un tel comportement pouvait être attendu, mais en dépit de cette légère sensibilité, la méthode apparaît robuste car les mêmes résultats sont trouvés même avec de petites variations structurales dans la structure requête.

6.2.4 Cas des protéines multi-domaines

Deux protéines multi-domaines ont été soumises aux serveurs de comparaison structurale : 2src, une kinase src humaine et 2hck (chaîne A), une kinase hck humaine. Les deux ont 4 domaines dans CATH : un domaine transférase, un domaine SH3, un domaine phosphorylase kinase et un domaine adaptateur SHC. Tous les 4 domaines peuvent être trouvés dans d'autres protéines, ou par paire ou un par un. C'est pourquoi, dans les résultats des serveurs, les structures partageant tous les 4 domaines devraient être classées premières, suivies par les structures possédant 2 des 4 domaines, elles mêmes suivies par les structures n'ayant qu'un de ces domaines. Pour DALI, VAST, MATRAS et CE, les structures similaires trouvées sont bien classées dans cet ordre. TOP et DEJAVU trouvent des structures partageant plus d'un domaine avec les requêtes (mais aucune avec un seul domaine) ; SSM trouve seulement des protéines avec les 4 domaines. LOCK trouve seulement des protéines représentatives pour chaque domaine unique des src kinases, mais pas de structure partageant tous les domaines. TOPSCAN et PRIDE sélectionne une seule structure avec un domaine kinase, et TOPS ne

trouve aucune structure similaire.

YAKUSA trouve beaucoup de structures similaires avec les 4 domaines, 2 domaines ou 1 domaine. Les structures avec 4 domaines sont classées premières, mais quelques structures partageant 2 domaines sont mieux classées que certaines à 4 domaines. Comme le score de classement n'est pas basé sur la longueur de l'alignement et que les alignements sont locaux, si les deux domaines d'une protéine sont plus similaires aux domaines de la requête que les 4 domaines d'une autre protéine, la protéine avec seulement deux domaines communs peut donner un meilleur score. C'est le but d'un alignement structural local de donner la priorité à des similarités locales fortes plutôt qu'à des similarités globales plus floues. Le comportement de YAKUSA pour les protéines multi-domaines est tout à fait satisfaisant.

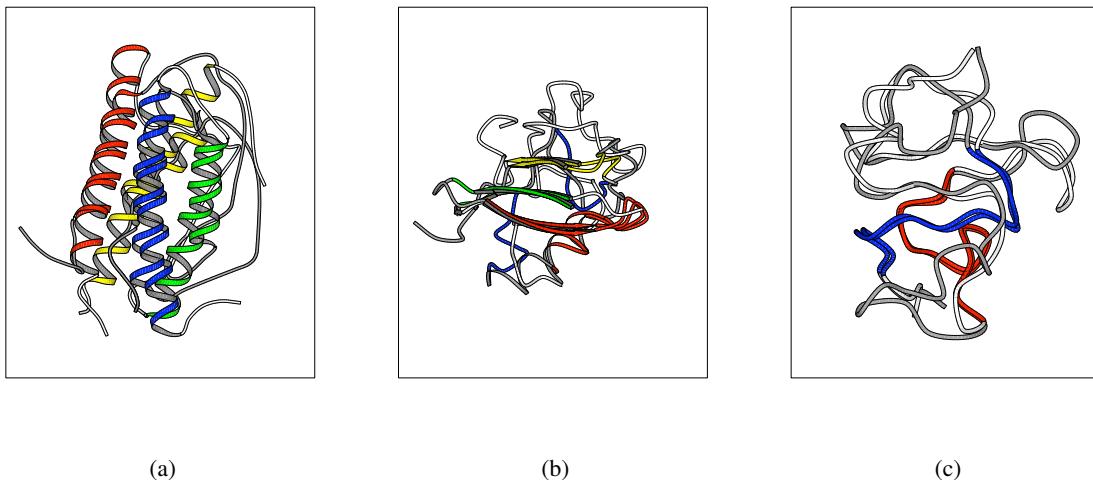


FIG. 6.12 – Alignements structuraux effectués par CE (figure a) et YAKUSA (figures b et c).

(a) Alignement structural par CE de deux structures de facteurs de stimulation des colonies de macrophages granulocytaires : 1bge (B), en blanc, et 2gmf (A), en gris. Les deux appartiennent à la même classe topologique CATH (*four helix bundle*). Les quatre hélices sont colorées. Sept résidus manquent dans 1bge : il y a un trou dans la boucle à droite de l'hélice verte. YAKUSA ne peut trouver ce type de similarités : l'orientation globale des quatre hélices α diffère puisqu'elles n'ont pas la même longueur et que les boucles sont différentes aussi. Le seul moyen de détecter ces deux structures comme similaires est d'aligner globalement les 4 hélices.

(b) Alignement par YAKUSA de la protéine de transfert d'électrons Fer-Soufre (1fxi chaîne A, blanche) et de l'ubiquitine (1ubq, grise). Elles appartiennent à la même classe de CATH (*mixed α – β , Roll, Ubiquitin like (3.10.20)*). Les quatre SHSP trouvés par YAKUSA sont colorés et les structures sont alignées selon le SHSP le plus long (colorié en rouge). Le SHSP bleu n'est pas spatialement compatible avec les autres.

(c) Alignement par YAKUSA de deux protéine de transfert d'électrons Fer-Soufre (1isu chain A (blanche) and 2hip chaîne A (grise)). Les deux structures appartiennent à la même classe de CATH : *few SSE, Irregular, High-potential iron-sulfur protein* (4.10.490). Les deux SHSP trouvés par YAKUSA sont colorés en rouge et bleu, et les structures sont alignées selon le SHSP le plus long (colorié en rouge). Les autres régions des deux protéines semblent similaires au premier coup d'œil mais elles sont trop courtes pour former des SHSP significatifs.

6.2.5 Cas non triviaux

Fisher et coll.(Fischer et al., 1996) ont sélectionné 68 paires de protéines qui ont des replis similaires mais qui ne sont pas facile à détecter, et ils ont calculé un niveau de difficulté pour ces paires. Nous avons sélectionné les 14 paires les plus difficiles, réparties dans les 4 classes principales

de CATH : 4 paires par classe sauf pour la classe *few SSEs* dans laquelle il n'y avait que 2 cas difficiles. Nous avons soumis les 14 requêtes à DALI et CE, qui sont les programmes les plus performants d'après les résultats ci dessus. Les résultats sont présentés dans le tableau 6.6.

TAB. 6.6 – Performances des programmes pour 14 cas non triviaux extraits de l'article de Fisher et coll. (Fischer et al., 1996).

Programme	<i>Mainly α</i>	<i>Mainly β</i>	<i>Mixed α – β</i>	<i>Few SSEs</i>	total
number of structures	4	4	4	2	%
YAKUSA	1	2	2	2	50%
CE	4	2	0	1	50%
DALI	2	2	0	0	29%

YAKUSA et CE ont les mêmes performances globales (50% de succès) mais la distribution des succès est différentes pour les deux programmes, et DALI échoue dans plus de cas (29% de succès). YAKUSA échoue pour trois paires de la classe *Mainly α* alors que CE trouve les quatre paires et que DALI en trouve deux. Cela montre un écueil de l'alignement local : puisque il y a beaucoup d'hélices α dans les structures de protéines et que les hélices sont très similaires, identifier deux repliements similaires partageant seulement des hélices α est difficile. Avec un alignement global, puisque ces hélices ont la même organisation globale, il est possible de trouver les deux structures similaires (figure 6.12(a)) alors qu'un alignement local ne peut identifier cette organisation globale. De plus, dans ce cas non trivial, les hélices α sont de longueurs différentes. D'un autre côté, YAKUSA se comporte mieux que CE et DALI pour des structures de la classe *mixed α – β* de CATH et pour la classe *Few SSEs*. YAKUSA trouve deux des 4 paires de structures *mixed α – β*, alors que CE et DALI n'en trouvent aucune. De plus, YAKUSA trouve les deux paires *Few SSEs*, CE n'en trouve qu'une et DALI n'en trouve pas. Des exemples de ces résultats sont présentés dans les figures 6.12(b) et 6.12(c). Les SHSP trouvés par YAKUSA sont souvent court mais leur $RMSD_c$ sont bas et les blocs structuraux trouvés sont très similaires. Si les SHSP ne sont pas constitués de SSE seulement, leur probabilité d'occurrence est faible et ils ont ainsi un bon score, et la compatibilité spatiale permet de différencier les vraies similarités des aléatoires. Cette méthode permet donc de trouver des structures distantes sans dépendre de la composition en SSE ni de la longueur des protéines.

Ce test montre aussi que DALI semble moins précis pour trouver des relations structurales lointaines que YAKUSA ou CE, et que ces deux derniers se montrent en quelque sorte complémentaires pour cette tâche importante. Comme attendu, CE est meilleur pour les similarités faibles mais globales (manquées par YAKUSA) et YAKUSA est meilleur pour les fortes similarités locales - non aléatoires - manquées par CE.

6.2.6 Détection des nouveaux repliements

Pour tester la capacité des méthodes à discriminer les nouveaux repliements (*i.e.* inconnus dans la banque), Novotny et coll. ont soumis aux serveurs quatre protéines classées comme singlenton,

i.e. elles sont uniques dans leur classe topologique de CATH. Le test est donc de ne pas trouver de *hit* signifiant. Trois de ces protéines test ne sont plus utilisables, puisque deux d'entre elles - 1cii and 1pya - ne sont plus des singletons dans CATH, et une troisième - 1pmi - possède un vrai voisin structural qui n'est pas classé dans CATH. Ainsi, la comparaison entre YAKUSA et les programmes de l'étude de Novotny ne serait pas adaptée (dans cette étude, les meilleurs résultats étaient obtenus par VAST qui trouvait la requête seulement dans deux cas et un autre *hit* avec un mauvais score dans deux cas). YAKUSA montre clairement une chute dans les Z-scores entre les *hits* pertinents et les non pertinents, sauf pour 1cii qui est une protéine transmembranaire constituée seulement d'hélices α . Le quatrième cas (1bgf) est encore un singleton et n'a pas de voisin structural selon les résultats donnés par YAKUSA : le Z-score tombe brutalement après le premier *hit* qui est la requête elle-même (chute du Z-score de 18.8 à 6.7).

Pour pousser plus loin l'étude du comportement de YAKUSA quand les requêtes sont des nouveaux repliements, nous avons sélectionné 8 structures singletons dans CATH, au moins au niveau « topologique » (1jvr, 1g7dA, 1tl2A, 1by2, 1cby, 4mt2, 1pnB, 1jsuC). Vu qu'il n'y a pas de seuil dans YAKUSA autre que celui fixé par l'utilisateur, YAKUSA peut toujours donner autant de résultats qu'on veut. Néanmoins, détecter que ces requêtes étaient des singletons a été facile car dans tous les cas, le Z-score chute brutalement après le premier *hit* - qui est la requête elle-même (les chutes vont de au dessus de 35 à environ 7 (minimum 5, maximum 10)). Ainsi, si le premier *hit* de YAKUSA est de l'ordre de 7, la requête a de fortes chances d'être un nouveau repliement.

6.3 Serveur

Avec l'aide de Sophie Brouillet, le service Yakunet (YAKUSA sur le web) a été écrit et intégré dans la plate-forme Ressource Parisienne en Bioinformatique Structurale (RPBS) qui permet l'accès à plusieurs ressources originales en Bioinformatique structurale (Allard et al., 2005) (voir figure 6.13). Il est accessible à l'adresse <http://www.rpbs.jussieu.fr/Yakusa>.

6.4 Conclusion

Comparé aux programmes similaires, YAKUSA se montre efficace : il est très rapide et peut trouver des similarités distantes. YAKUSA permet une recherche en temps réel sur des banques de structures sans avoir besoin de simplifier à outrance les structures ou de pré-calculer les résultats. Il peut donc traiter les nouveaux repliements aussi efficacement que des repliements similaires à ceux déjà connus. De plus, YAKUSA détecte des similarités manquées par d'autres programmes parce qu'il ne s'appuie pas sur les SSE et qu'il trouve des similarités fortes et locales au niveau des C_α . Il se comporte au moins aussi bien que les meilleures méthodes dans les cas usuels. Il semble mieux adapté pour détecter des structures distantes dans les familles *mixed $\alpha - \beta$* ou *Few SSEs* que CE ou DALI alors que CE semble mieux se comporter pour les cas de structures distantes dans la classe *Mainly α* .

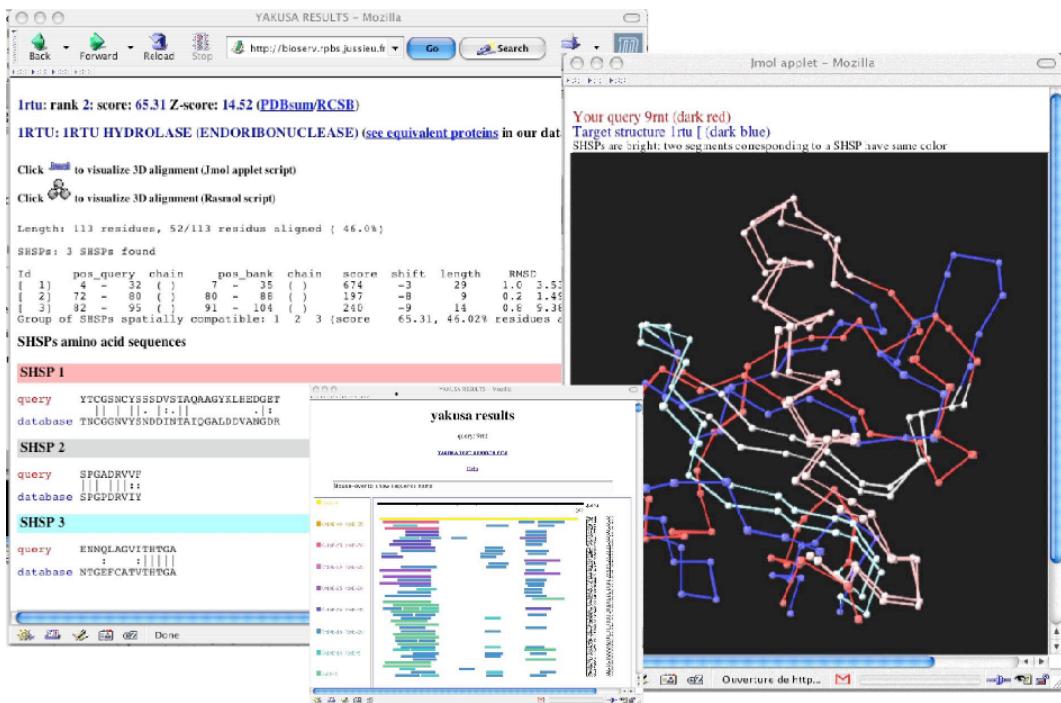


FIG. 6.13 – Exemple de pages de résultats de YAKUSA : La fenêtre au premier plan est une liste synthétique des résultats de la recherche de structures similaires dans la banque. Pour chaque structure trouvée, les SHSP sont colorés suivant leur score. La fenêtre graphique à droite est la superposition de deux des structures (9RNT en rouge, la structure requête et IRTU en bleu foncé) selon le SHSP rose. Les deux autres SHSP trouvés sont en bleu clair et en gris (ici, la couleur des SHSP permet de les différencier mais n'a pas d'autre signification). La fenêtre à gauche est le résultat détaillé de l'alignement de ces deux structures.

La méthode implémentée dans YAKUSA combine les avantages de la rapidité, de la recherche de similarités locales et d'un classement basé sur les probabilités d'occurrence des segments structuraux trouvés. La rapidité de la méthode permet des expérimentations de comparaison « tout contre tout ». C'est pourquoi nous l'avons utilisé comme première étape dans la procédure d'obtention d'une classification automatique de la PDB.

CHAPITRE 7

Comparaison de multiple de structures

Dans le but de constituer les coeurs des familles de protéines, nous avons développé plusieurs méthodes de recherche de motifs structuraux dans un ensemble de structures en vue d'alignements multiples de structures. Trois méthodes seront développées ici, deux sont des extensions d'algorithmes déjà connus dans le domaine de l'analyse de séquences mais modifiés pour les spécificités de l'analyse des structures de protéines : la méthode des *m*-diagonales, une variante du "Gibbs sampling", la troisième méthode est une extension de l'algorithme KMRC (Soldano et al., 1995) à la recherche de "motifs relationnels flous". Cette méthode a une application plus générale, mais son application à la recherche de motifs structuraux en montre bien les capacités.

7.1 Méthode des *m*-diagonales

7.1.1 Méthode

La première méthode que nous avons développée est inspirée d'une méthode d'alignement multiple de séquences biologiques, MACAW (Schuler et al., 1991; Lawrence et al., 1993; Karlin and Altschul, 1990). Cette méthode permet de trouver des blocs sans gap similaires dans les séquences biologiques.

Elle comprend deux grandes étapes : i) aligner toutes les paires de séquences, les alignements déterminés formant des « diagonales de similarité » (2-diagonales) ; ii) combiner les 2-diagonales pour obtenir des *m*-diagonales (voir la figure 7.1), *m* étant le quorum de protéines voulues dans un bloc.

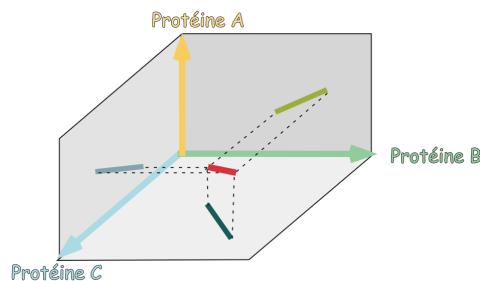


FIG. 7.1 – Représentation schématique de la méthode des *m*-diagonales : trois protéines sont alignées (A, B, C) donnant lieu à trois 2-diagonales (en deux dimensions). La combinaison de ces trois 2-diagonales forme une 3-diagonale en rouge (en trois dimensions). Le quorum est ici de 3 (le bloc est présent sur toutes les protéines).

Comme notre représentation des structures protéiques est linéaire (suite d'angles α), il est possible d'adapter cet algorithme pour l'étendre à la recherche de motifs structuraux.

7.1.2 Adaptation aux structures

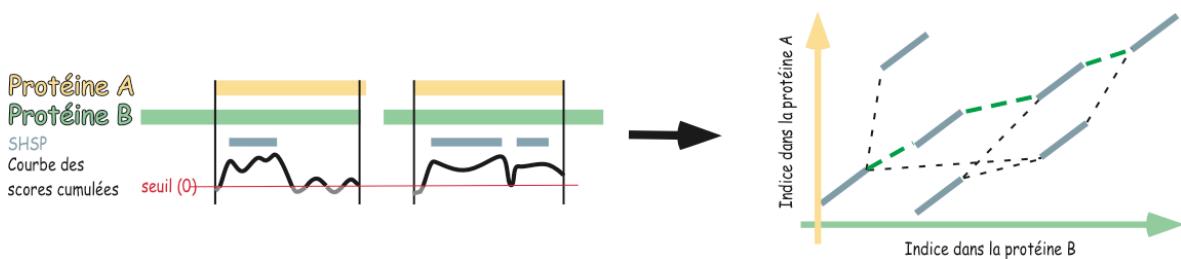


FIG. 7.2 – Alignement d'une des paires de structures.

À gauche : Les deux traits jaune et vert sont les deux structures. Les traits gris sont les sous-segments de score maximaux (scores angulaires voir texte). Le score cumulé est la somme des scores S_{sim} de chaque paire de résidus tant que cette somme est positive. Deux décalages sont présentés. Ici, trois segments de score maximaux sont trouvés.

À droite : Parmi les 7 SHSP (2-diagonales) trouvées, seul quatre sont inclus dans la série de meilleur score (chemin en vert).

Première étape : trouver les 2-diagonales (aussi nommées SHSP).

Les structures protéiques sont converties en suites d'angles α . Le meilleur alignement entre deux structures est déterminé en recherchant les sous-segments de score maximaux pour tous les décalages possibles entre deux structures. Pour un décalage donné, un vecteur de score S_{sim} est calculé pour toutes les paires d'angles α selon la formule déjà utilisée dans YAKUSA $S_{sim} = T - D_a(\alpha_i, \alpha_j)$ où $D_a(\alpha_i, \alpha_j)$ est la différence angulaire entre deux angles α_i et α_j de la paire de structures concernées - alignés pour un décalage donné -, et T une constante permettant d'obtenir un score entre deux angles en moyenne négatif pour des angles aléatoires. De ce vecteur de scores ne sont conservés que les sous-segments de score maximaux (SHSP en gris dans le schéma 7.2). Ces sous-segments sont détectés en sommant les scores dans l'ordre de la séquence. Lorsque la somme des scores devient négative, elle est mise à 0 et le sous-segment ayant atteint le score maximal est conservé (voir figure 7.2, gauche). Ensuite, seule la meilleure suite de sous-segments (SHSP) détectés est conservée (celle ayant le meilleur score ou la plus faible probabilité, voir figure 7.2, droite). Pour cela, toutes les séries possibles de SHSP sont énumérées et leurs scores calculés. Pour calculer la probabilité d'une série, le même modèle MTD que celui de YAKUSA est utilisé. Le score d'une série est alors soit sa probabilité soit la somme des scores S_{sim} des 2-diagonales qui la composent.

Soit m structures, chacune de longueur ℓ . L'alignement de deux structures pour un décalage donné est en $O(\ell)$ et il y a 2ℓ décalages pour une paire de structures. L'alignement des $m(m-1)/2$ paires de structures est donc en $O(m^2\ell^2)$. La recherche de la meilleure série de 2-diagonales pour une paire de protéines est en théorie en $O(\ell^2!)$ car il y a un nombre proportionnel à ℓ de 2-diagonales par décalage, 2ℓ décalages et $x!$ séries possibles de x éléments. Cependant, nous avons fixé un nombre maximum M_{max} de 2-diagonales par alignement. Par conséquent, la recherche de la meilleure série de 2-diagonales est bornée par $M_{max}!$ et donc effectuée en temps borné (et court...).

Seconde étape : combiner les 2-diagonales en *m*-diagonales

La combinaison des alignements n'est pas réalisée - comme dans beaucoup d'autres méthodes - de manière hiérarchique mais par construction de graphes comme dans l'article original de G. Schuler et coll. (Schuler et al., 1991). Les sommet de ces graphes sont les résidus et une arête est définie lorsque deux résidus sont alignés dans une 2-diagonale. Les graphes connexes sont ensuite recherchés. Dans le cas le plus simple, une seule position par structure est présente dans un graphe connexe (graphes G1 et G2 de la figure 7.3) mais dans certains cas, les graphes sont plus complexes (graphe G3 de la figure 7.3). Les blocs sont ensuite construits en parcourant ce graphe et la meilleure série de blocs est recherchée de la même manière que pour l'alignement de deux structures. L'algorithme général est détaillé page 128 (algorithme 3).

7.1.3 Algorithme

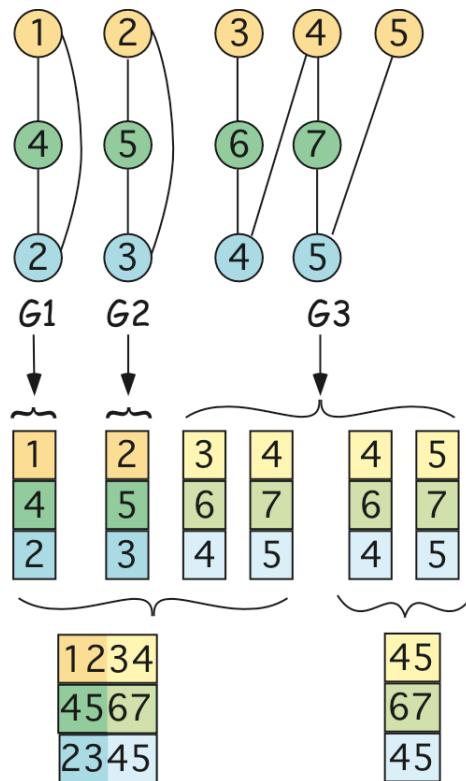


FIG. 7.3 –

Algorithm 3 Algorithme général d’alignement des structures selon la méthode des m-diagonales.

```

//Lecture des arguments et des fichiers d’entrée (les structures protéiques)
//Alignement des toutes les paires de structures
TrouveTousAli22(tProtéines, paramètres);
//Construction du graphe
ConstruireGraphe(tProtéines, paramètres);
for all protéine dans tProtéines do
    tPilesAli ← ParcoursGrapheProf(protéine, tProtéines, paramètres)
end for
//Le graphe est parcouru à partir de chaque protéine
//Construction des blocs à partir des positions adjacentes
pPileBlocs ← TrouveBlocs(tPilesAli, paramètres, tProtéines);
//Recherche de la meilleure série de blocs
pCheminBlocs=TrouveCheminBlocs(pPileBlocs, paramètres, tProtéines);
//Écriture des fichiers de sortie

```

Construction du graphe des 2-diagonales

L’algorithme de construction du graphe est une implémentation de l’algorithme de R. Sedgewick (Sedgewick, 1991). Le graphe construit est peu dense car chaque sommet, qui est un résidu d’une structure, ne peut être lié/aligné au maximum qu’à un résidu de chaque structure. Ce graphe a donc au maximum $m - 1$ arêtes, m étant le nombre de structures. La représentation d’un graphe peu dense la mieux adaptée est la représentation par structure d’adjacence, sauf s’il faut pouvoir facilement supprimer des sommets ce qui n’est pas notre cas. Une liste d’adjacence contient un lien vers tous les sommets auxquels un sommet est lié. Chaque sommet a une liste d’adjacence contenant tous les

sommet avec qui il partage une arête. Les sommets du graphe sont stockés dans un arbres binaires de recherche. Un arbre binaire est créé pour chaque protéine.

Au maximum, le graphe construit a autant de sommets qu'il y a de résidus dans les structures, donc $m\ell$ sommets. La création d'un sommet dans le graphe revient à ajouter un élément dans un arbre binaire. Il faut donc en moyenne $2\ln(\ell)$ comparaisons pour créer un sommet. La création de tous les sommets est donc de complexité moyenne $O(\ln(\ell)m\ell)$ et dans le pire des cas en $O(m\ell^2)$ (si l'arbre est complètement déséquilibré).

Comme il y a $m\ell$ sommets chacun lié au maximum à $m - 1$ sommets, il y a $m^2\ell$ arêtes au maximum. Etant donné ce maximum de $m - 1$ arêtes par sommet, les listes d'adjacence sont très courtes et sont donc représentées par des tableaux. L'ajout d'une arête à une liste est fait en temps constant. Trouver le sommet auquel ajouter l'arête revient à chercher le sommet dans l'arbre binaire de recherche, la complexité moyenne est donc en $O(\ln(\ell))$. Néanmoins, la complexité dans le pire des cas de l'insertion de toutes les arêtes est en $O(m^2\ell^2)$ ²³.

Parcours du graphe

Les composantes connexes du graphe contiennent les alignements multiples pour chaque résidu. Elles sont donc recherchées en parcourant le graphe en profondeur d'abord. Il est montré dans la figure 7.3 que dans certains cas une composante connexe peut être convertie en plusieurs alignements de résidus.

Constructions des blocs

Les alignements multiples de taille 1 (« alignements unitaires ») déterminés à l'étape précédente sont regroupés pour former des alignement sans gap (blocs) de longueur supérieure si tous les résidus alignés sont contigus (voir figure 7.3). Seuls les meilleurs blocs sont conservés.

Recherche de la meilleure série de blocs

La recherche de la meilleure série de blocs est effectuée comme pour les alignements de paires de protéines mais en m dimensions (m étant le nombre de structures). Tout comme pour la recherche des meilleures séries de 2-diagonales, le score d'une série est soit sa probabilité soit la somme des scores S_{sim} . Cependant, il faut prendre en compte la connectivité des graphes des alignements unitaires qui ont formé le bloc.

7.1.4 Paramètres et temps d'exécution

Cette méthode a été implémentée en langage C. Le temps d'exécution dépend évidemment du nombre de protéines à aligner (voir figure 7.4).

7.1.5 Exemples

²³La création des sommets et l'insertion des arêtes ont lieu en même temps.

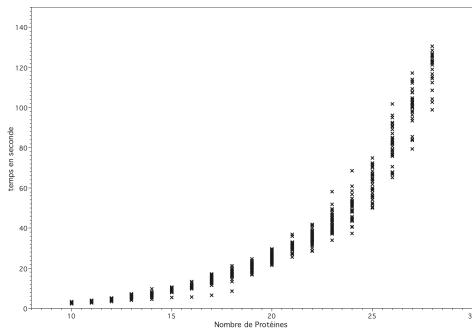


FIG. 7.4 – Temps d'exécution en fonction du nombre de protéines à aligner (le quorum est fixé au nombre de protéines à aligner).

Cette méthode a été utilisée pour aligner des familles de structures qui seront présentées par la suite. Nous avons choisi d'étudier des structures de la superfamille multigenique des cytochromes P450 (CYP, P450). Ce sont des protéines impliquées dans beaucoup d'oxydations de substrats hydrophobes. Les substrats sont des hormones stéroïdes, acides gras, molécules signales extracellulaires et des vitamines mais également des substrats exogènes comme drogues ou polluants environnementaux (voir le (Estabrook, 2003) pour une revue historique). Ces P450s peuvent être trouvés dans beaucoup les êtres de vivants : bactéries, levure, champignons, végétaux, insectes, poissons et mammifères. Ils ont été largement étudiés notamment en raison de leur rôle dans la dégradation de drogues. Leurs séquences primaires d'acides aminés sont différentes malgré leurs similitudes structurales. Les alignements obtenus pour 35 cytochromes P450 sont présentés ci-après.

Si le quorum est réduit à 10 protéines, la longueur moyenne d'un alignement passe à 143 résidus répartis en 13 blocs. Les 4 blocs contenant les 35 protéines sont toujours présents.

7.1.6 Conclusion

Une limitation de cette méthode est que l'alignement multiple final est généré à partir des alignements optimaux de paires de structures. Or un alignement multiple optimal n'est pas obligatoirement composé d'alignements optimaux de paires de structures. Pour pallier, en partie, cette limitation, il serait possible de conserver non pas un alignement de 2-diagonales pour chaque paire de structures, mais toutes les 2-diagonales ayant, par exemple, un score suffisant. Cependant, l'algorithme tel qu'il a été décrit - et implanté - ne supporterait pas cette limitation. En effet, chaque sommet (résidu d'une protéine) pourrait alors être lié (aligné) avec plusieurs sommets d'une même autre protéine.

Cette méthode a plusieurs avantages. Elle est assez rapide et adéquate pour des alignements de quelques dizaines de protéines (moins de 3 mn sur un Macintosh G5 pour 30 protéines de 400 résidus). Toutes les protéines ne sont pas nécessairement alignées (si le quorum n'est pas au maximum). Ainsi, si une protéine est très différente des autres, elle ne sera pas dans l'alignement. Comme ce cas peut survenir dans des familles de protéines générées automatiquement - ce que nous cherchons à faire -, c'est un avantage important. Enfin, c'est une méthode d'alignement local et sans *a priori*, donc

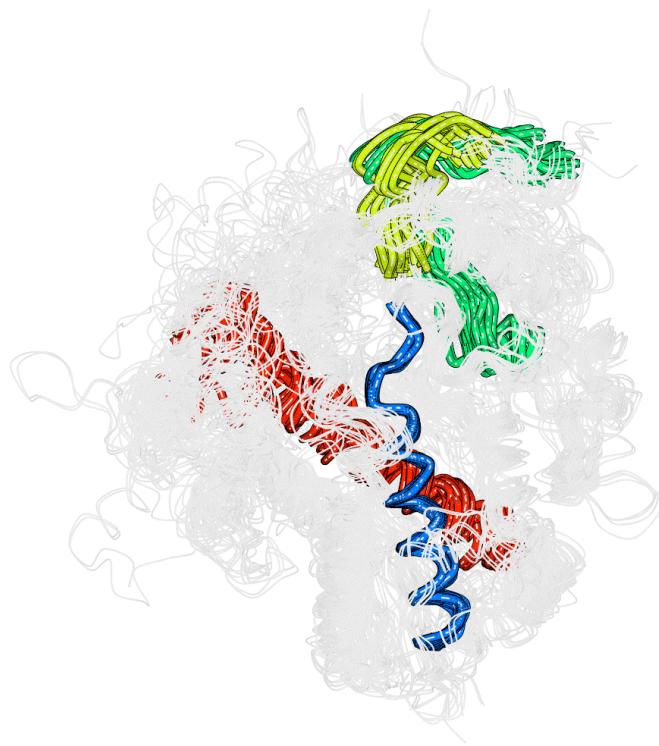


FIG. 7.5 – Alignement structural par la méthode des *m*-diagonales de 35 chaînes de cytochromes P450 (origines : bactéries, champignons, mammifères). Toutes les protéines devaient être présentes dans chaque bloc (quorum maximal). La longueur total de l'alignement est de 81 résidus par protéines en 4 blocs. Les protéines ont en moyenne 412 résidus. Les structures sont superposées selon le bloc bleu. Les autres blocs semblent donc moins similaires mais il s'agit seulement de l'effet « superposition » locale du bloc bleu qui « décale » les autres blocs (cf. figure 7.6)

adaptée à la détermination de cœurs structuraux.

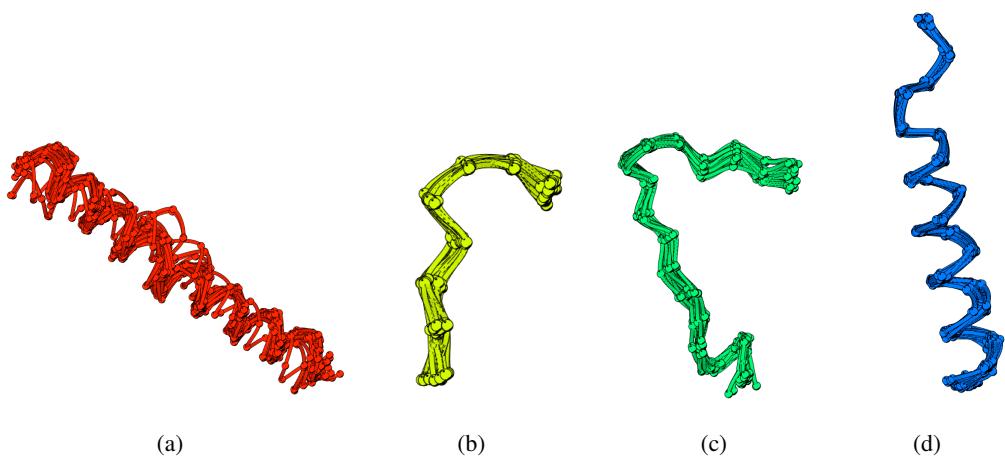


FIG. 7.6 – Les quatre blocs structuraux trouvés pour les 35 cytochromes sont ici représentés séparément.

7.2 Méthode du Gibbs Sampling appliquée aux structures

7.2.1 Méthode générale

L'algorithme de *Gibbs Sampling* est aussi un algorithme qui a été adapté au départ pour les séquences biologiques (Lawrence et al., 1993; Neuwald et al., 1995; Thompson et al., 2003). Il permet de trouver les motifs répétés flous dans un ensemble de séquences. Nous avons adapté cet algorithme pour chercher un motif commun dans les structures. Comme dans les méthodes précédentes, nous avons représenté les structures par une série d'angles α qui sont discrétilisés.

Exemple pour $m = 3$ structures, $\beta = 1,41$; $l = 4$; il y a quatre "angles" : 

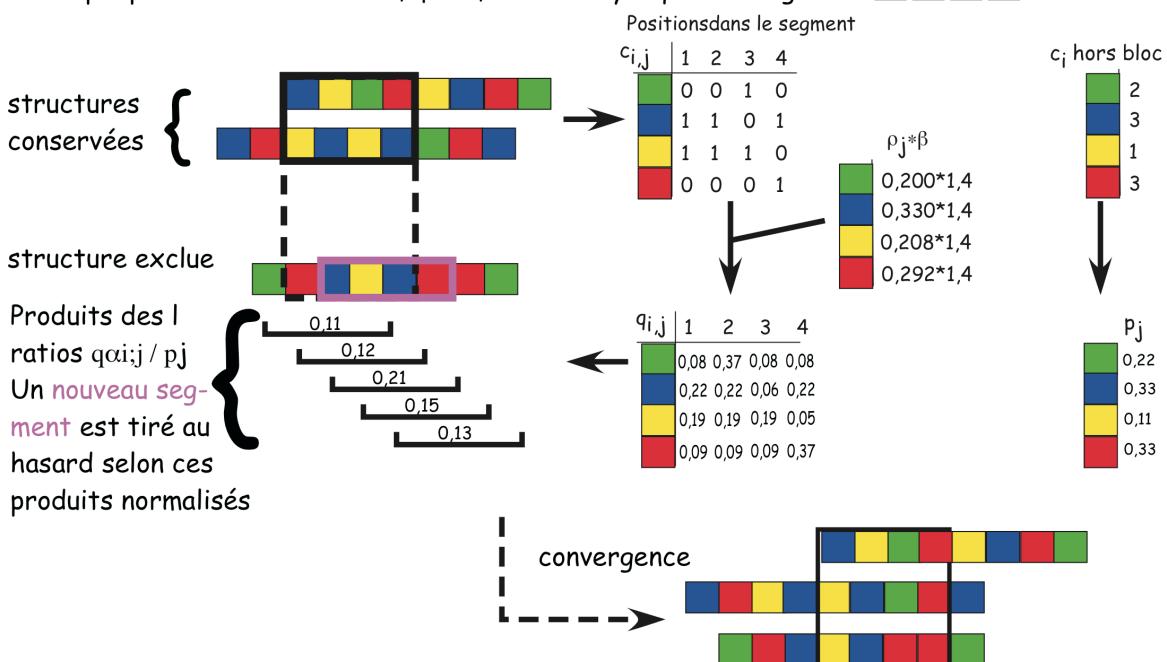


FIG. 7.7 – Exemple simplifié de l'algorithme de Gibbs sampling. Les différents symboles sont les 4 couleurs (rouge, jaune bleu et vert). Trois structures sont alignées et des blocs de taille l sont recherchés.

On recherche un bloc structural similaire de taille l donnée dans m structures recodées en angles α discrétilisés. L'algorithme suivant est répété plusieurs fois. Un segment de taille l est choisi au hasard dans chacune des m structures. Puis, les deux étapes suivantes sont répétées jusqu'à convergence.

1. Une structure est exclue et les probabilités de chaque angle α_i à chaque position j dans les segments des $m - 1$ autres structures sont calculées selon la formule :

$$q_{i,j} = \frac{c_{i,j} + \rho_i \cdot \beta}{m - 1 + \beta} \quad (7.1)$$

où i représente l'angle α discrétilisé et j la position dans le bloc, m est le nombre de structures, $c_{i,j}$ le nombre d'angles α_i et similaires à α_i en position j des segments, ρ_i la fréquence de

base des angles α_i dans toutes les structures et β une constante (généralement prise égale à $\sqrt{m - 1}$). Le produit $\rho_i \cdot \beta$ représente les *pseudocounts* pour l'angle α_i . Plus cette constante β est élevée par rapport à m et plus les *pseudocounts* auront un poids élevé. De manière générale, un pseudocount est un comptage additionné à une donnée pour changer sa probabilité - qu'on sait être non nulle - afin qu'elle soit négligeable mais pas nulle (on suppose que la non observation de la donnée provient de l'échantillonnage trop petit). Les fréquences p_i des α_i hors des segments sont aussi calculées selon cette formule.

2. Dans la structure exclue précédemment, chaque position k possible du motif est évaluée en calculant Q_k/P_k , le produit des l ratios $\frac{q_{i,j}}{p_i}$ (i représente l'angle α_i se trouvant en position j , $k < j < k + l - 1$). Ces évaluations sont normalisées en probabilités selon lesquelles on tirera au hasard un de ces segments, qui devient la nouvelle position du motif pour cette structure. Ce segment sera utilisé pour calculer les $q_{i,j}$ de l'étape 1 lors de l'exclusion d'une autre structure. De manière équivalente, on peut maximiser F , la somme des logarithmes de ces ratios :

$$F = \sum_{i=1}^{\text{Nb symboles}} \sum_{j=1}^l \frac{q_{i,j}}{p_j} \quad (7.2)$$

Dans la méthode originale de Gibbs sampling décrite ci-dessus, seule l'identité des angles (ou symboles) est prise en compte et non la similarité ; pour les structures, ceci conduit à une estimation trop stricte et donc infructueuse des blocs structuraux similaires. Nous avons donc ajouté dans l'étape 1 un terme supplémentaire de *pseudocounts* basé sur la similarité entre les angles : lorsqu'un angle est compté à une position donnée, une fraction de ce comptage est aussi ajouté aux angles proches. Cette fraction dépend d'une gaussienne centrée autour de l'angle et d'écart type choisi ; plus l'écart type est grand, plus cette fraction sera grande pour les angles moins proches de l'angle considéré (donc plus on considérera comme importante la similarité entre angles moins proches). On calcule donc les $q_{i,j}$ de l'étape 1 comme suit :

$$q_{i,j} = \frac{\sum_{k=\text{angles}} \gamma_{j,k} \cdot c_{i,k} + \rho_i \cdot \beta}{m - 1 + \beta} \quad (7.3)$$

où $c_{i,k}$ est toujours le nombre d'angles α_k en position i et γ_{ik} est la probabilité de l'angle discréétisé α_k dans une gaussienne - discréétisée aussi (figure 7.8) - de moyenne α_i et d'écart type choisi.

Cet algorithme est sensible au décalage, c'est-à-dire qu'un minimum local peut être trouvé qui chevauche seulement un vrai motif. Une heuristique utilisée (identique à celle de la méthode originale) est de « décaler » la position du motif trouvé afin de vérifier qu'un motif décalé par rapport au motif trouvé n'a pas un meilleur score (en ce cas, on garde le motif décalé).

Une autre difficulté est que le score optimal F , augmente avec la longueur du motif. Et une limitation de la méthode est de devoir fixer la longueur des blocs recherchés. Un critère, G , soustrait à la fonction F l'information requise pour déterminer la position du motif dans chacune des séquences. Il est appliqué aux résultats et prend en compte le score trouvé, F , le nombre de degrés de liberté utili-

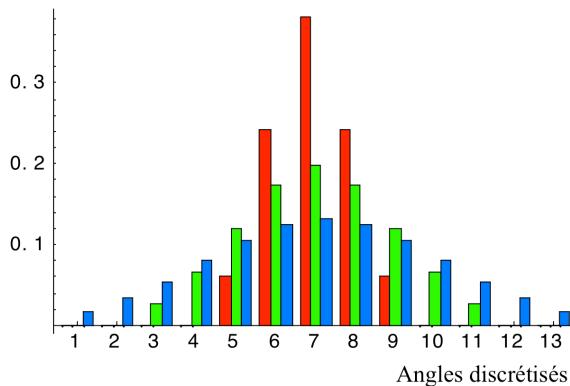


FIG. 7.8 – Gaussienne discréétisée : la moyenne est ici à 7. En rouge, les coefficients pour un écart type de 1, en vert, pour un écart type de 2 et en bleu pour un écart type de 3.

sés. Si on divise G par la longueur du motif multipliée par le nombre de symboles dans le vocabulaire moins un - 35 pour notre alphabet d’angles, 19 pour les acides aminés dans les séquences), on obtient un score que Lawrence appelle l’information par paramètre. Ce score atteint un maximum pour une longueur de motif optimale et chute après. Ce critère G est calculé comme suit :

$$G = F - \sum_{s=1}^m \left(\log L'_s + \sum_{k=1}^{L'_s} (Y_{s,k} \log Y_{s,k}) \right) \quad (7.4)$$

où L'_s est le nombre de positions possibles pour le motif dans la séquence s , et $Y_{s,k}$ est le poids normalisé de la position k , c’est-à-dire le poids Q_k/P_k divisé par la somme de ces poids dans la séquence s . Cet ajustement tient compte du fait que la position du motif sur chaque séquence n’est pas connu. Ceci permet une comparaison des scores de motifs de longueurs différentes. Notamment, une phase de contraction-expansion autour du motif trouvé après convergence permet d’ajuster la longueur la plus pertinente pour ce motif grâce à ce critère. De plus, l’algorithme entier est réitéré avec des longueurs différentes et en masquant les blocs déjà trouvés.

Evidemment, la méthode applique plusieurs fois l’algorithme pour trouver plusieurs motifs similaires. Ainsi, plusieurs segments structuraux communs sont en général trouvés pour chaque famille.

Cette méthode modifiée peut s’appliquer non seulement à des structures mais aussi à des séquences d’acides aminés, les *pseudocounts* supplémentaires sont alors basés sur les fréquences de substitution tirées d’une matrice de similarité de type BLOSUM ou PAM. Cette caractéristique permet d’améliorer la reconnaissance de blocs similaires difficiles à exhiber par le Gibbs sampling classique (on ajoute de l’« information biologique » quant à la « substituabilité » des acides aminés).

Dans la méthode, la procédure d’optimisation du *Gibbs Sampling* peut être combinée avec une recherche « tabou ». L’algorithme de recherche tabou (Glover, 1989) est une méta-heuristique simple dans laquelle on maintient une liste *first in - first out* des positions déjà examinées. Cette liste - de longueur choisie - permet de ne pas repasser par des positions déjà visitées, avec une certaine mémoire

(dépendant de la longueur de la liste).

7.2.2 Temps d'exécution

Le graphique 7.9 des temps d'exécution montre que celui augmente modérément avec le nombre de protéines mais qu'il est assez variable, ce qui est logique pour une méthode d'échantillonnage.

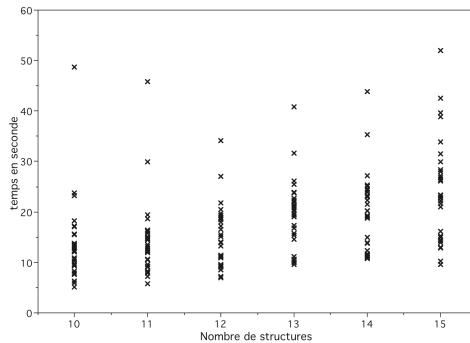


FIG. 7.9 – Temps d'exécution de la méthode de Gibbs sampling en fonction du nombre de protéines à aligner.

7.2.3 Exemples

Cette méthode, comme la précédente, a été utilisée pour aligner des familles de structures qui seront présentées par la suite et les résultats obtenus pour les cytochromes P450 sont présentés ci-après.

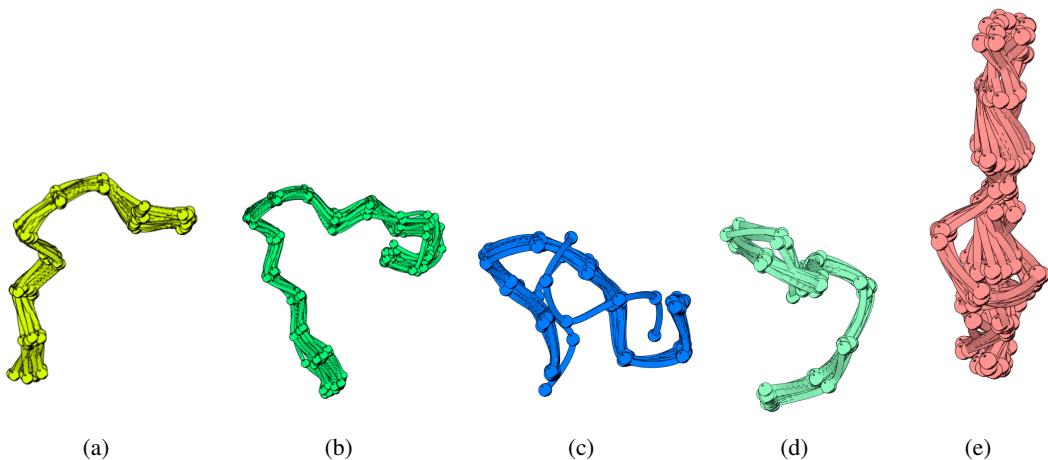


FIG. 7.10 – Les cinq blocs structuraux trouvés par la méthode du Gibbs sampling pour les 35 cytochromes sont ici représentés séparément. Les trois premiers blocs sont aussi trouvés par la méthode précédente des *m*-diagonales (les couleurs correspondent).

7.2.4 Conclusion

Cette méthode a pour désavantage que la longueur des motifs cherchés doit être fixée au départ, mais on a vu que cette longueur peut être variée et qu'un critère statistique permet de comparer des motifs de tailles différentes. De plus, on peut chercher plusieurs séries de motifs. La méthode n'a pas de quorum - le motif est cherché sur toutes les séquences - et un bloc peut donc contenir un fragment non similaire si la structure qui le porte n'en contient pas. Les avantages de la méthode sont qu'aucun alignement par paires préalable n'est nécessaire et que la méthode est locale. De plus, le temps d'exécution est moins dépendant du nombre de protéines que la méthode des m -diagonales et cette méthode permet de comparer un très grand nombre de structures.

7.3 Recherche de motifs répétés

7.3.1 Présentation générale

Nous avons vu dans l'état de l'art que l'algorithme de KMR a été adapté - sous forme de l'algorithme KMRC - à la recherche de motifs structuraux approchés par H. Soldano et coll. et par M.-F. Sagot et coll. (Soldano et al., 1995; Sagot et al., 1995; Jean et al., 1997) (voir page 76). Dans cette méthode, la description de la structure tridimensionnelle des protéines est réalisée soit par les angles (ϕ, ψ) , soit par les angles (α, τ) (cette description est linéaire). Certaines des meilleures méthodes de comparaison de deux structures, utilisent une description en distances internes des structures. Or il n'existe aucune méthode de comparaison multiple de structures travaillant avec les distances internes. Nous avons donc développé un algorithme, inspiré de l'algorithme de KMRC, permettant de comparer de manière multiple les structures selon leurs distances internes. L'implémentation de cette méthode est nommée Triades.

Les distances internes, comme les angles, peuvent être échantillonnées suivant un pas ou **maille** (en Angström) et donc être représentées par des symboles associés à chaque classe (intervalle). Cependant, les distances internes sont différentes des angles car un angle est associé à un résidu tandis que la distance est calculée entre deux résidus. Il s'agit donc d'une **relation** entre deux résidus et non simplement la caractéristique d'un unique résidu. Chercher des fragments structuraux similaires représentés par leurs distances internes revient à rechercher les fragments composés de résidus contigus dont les distances internes - les relations - sont identiques ou similaires (figure 7.3.1). Ces fragments sont appelés **motifs relationnels**.

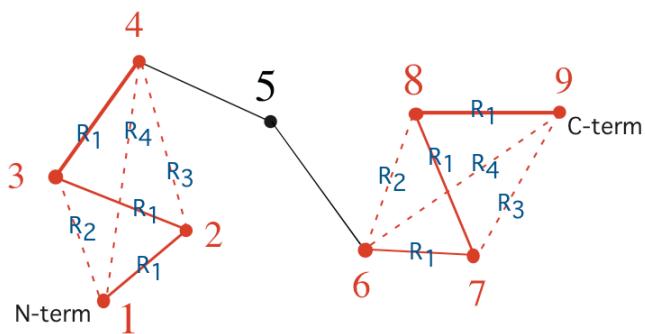


FIG. 7.11 – Exemple de fragments structuraux dont les distances internes sont proches et donc ayant des relations identiques (relations R_1 à R_4). Les points représentent les C_α . Les fragments structuraux proches sont en rouge, les lignes pointillées représentent les distances internes autres que celle entre C_α contigus. La distance entre deux C_α successifs varie peu, la relation est toujours R_1 . Par contre, les autres distances varient et les relations sont R_2, R_3, R_4 (en bleu).

L'algorithme décrit ici permet de trouver des motifs composés à la fois de symboles similaires et de relations similaires. Dans l'application au cas des structures, seules les relations sont prises en compte, le même symbole est donc affecté à tous les résidus. Cependant, d'autres applications de cet algorithme utilisent les symboles et les relations.

7.3.2 Quelques définitions

Le vocabulaire et les notations sont les mêmes que ceux utilisés lors de la description de KMRC. Pour rappel, la structure est représentée par une suite de symboles de l'alphabet Σ . Ce texte est nommé s et est de longueur n . Les symboles de l'alphabet sont regroupés en **pavés** recouvrants et cet ensemble de pavés est nommé **l'alphabet dégénéré**. La **dégénérescence symbolique** g_S est le nombre maximal de pavés de symboles auquel peut appartenir un symbole de l'alphabet Σ . Les motifs de longueur k sont des **k -motifs**. L'ensemble des **occurrences** d'un motif est nommé son **extension**. Un motif **répété** est un motif présent dans au moins q structures (quorum). Un k -motif est dit **maximal** si son extension n'est incluse dans aucune autre extension de k -motif.

Lorsque les relations sont prises en compte, le texte de départ est enrichi par les relations entre chaque paire de positions. Il faut remarquer qu'il est inutile de connaître les relations entre symboles séparés par plus de k symboles si l'on ne cherche que les motifs répétés de longueur k . L'ensemble des relations forme **l'alphabet relationnel** $R = \{r_1, \dots, r_{|R|}\}$. Le texte relationnel s est une chaîne de caractères de longueur n sur l'alphabet Σ où pour chaque paire (p_1, p_2) de positions, il y a une relation unique (symétrique) $r \in R$ entre les positions p_1 et p_2 qui pourra être notée $r(p_1, p_2)$. Ainsi, la quantité de données en entrée n'est plus n mais $n \times k$ (car il y a k relations par position, voir figure 7.3.2). Pour les relations comme pour les symboles un certain degré d'approximation est permis pour permettre la recherche de motifs non exacts. Cette approche est effectuée en découplant l'espace de alphabet relationnel en **pavés relationnels** recouvrants CR_i , dont l'ensemble ou **alphabet relationnel dégénéré** est noté G_R . Comme pour les pavés de symboles, les CR_i peuvent avoir des relations en communs (ils peuvent se chevaucher), mais aucun n'est inclus dans un autre. La **dégénérescence relationnelle** g_R est le nombre maximal de pavés de relations auquel peut appartenir une relation. Un **k -motif relationnel** est donc un k -motif de longueur k enrichi par les pavés relationnels définis entre chaque paire de positions. Il s'écrit donc avec les symboles de l'alphabet relationnel dégénéré ainsi que de l'alphabet dégénéré non relationnel. L'ensemble des positions d'un motif relationnel est toujours nommée une *extension* et un k -motif répété relationnel est dit maximal si son extension L_I n'est incluse dans aucune autre. Comme pour les motifs (non relationnels), une position peut appartenir aux extensions de plusieurs motifs relationnels.

Le problème posé peut alors se formuler :

Problème : Trouver les k -motifs répétés relationnels maximaux :

ENTRÉE : La séquence relationnelle s , les pavés G sur l'alphabet Σ , les pavés relationnels G_R sur les relations R , et la longueur k .

SORTIE : Les extensions des k -motifs répétés relationnels maximaux.

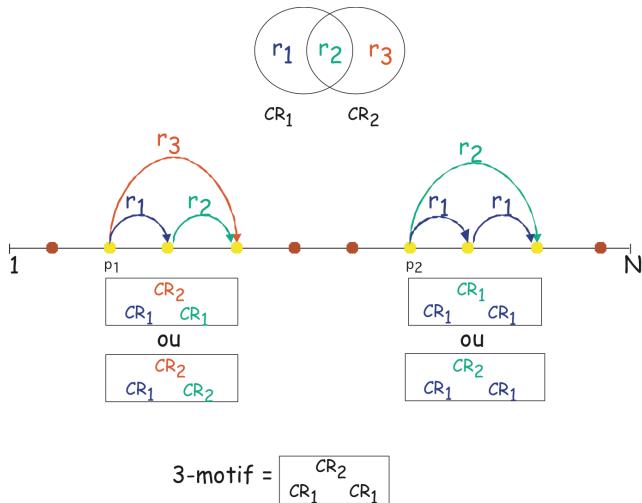


FIG. 7.12 – Exemple de deux motifs relationnels de taille 3. Trois relations sont définies : $R = \{r_1, r_2, r_3\}$ regroupées en deux pavés relationnels tels que $G_R = \{CR_1, CR_2\}$, $CR_1 = \{r_1, r_2\}$ et $CR_2 = \{r_2, r_3\}$. Les symboles (non relationnels), représentés par des points jaunes ou marrons, sont omis car ils sont tous identiques dans ces exemples (et dans les suivants aussi). Les deux instances de motifs relationnels sont différentes uniquement au niveau des relations. Comme la relation r_2 appartient à deux pavés relationnels, p_1 et p_2 , sont chacun dans les extensions de deux motifs différents. Cependant, un des 3-motif est le même pour les deux positions (le motif en noir), il est donc répété.

7.3.3 Construction des motifs

Dans KMRC, deux k -motifs permettaient de construire un motif de taille $2k$. Dans le cas des motifs relationnels, un tel mode de construction impliquerait la vérification de toutes les relations entre chaque position du premier k -motif et chaque position du second k -motif, donc k^2 relations. Il est alors plus judicieux de construire les motifs de taille supérieure à partir de deux k -motifs décalés de 1 (chevauchants sur $k - 1$). Un $k + 1$ -motif est alors construit et seule une relation est à vérifier : la relation entre la première et la dernière position soit la relation $r(p_i, p_{i+k})$ (voir figure 7.13.A). Soit un $k + 1$ -motif dont une instance est p , construit à partir des k -motifs dont une instance est p ou $p + 1$. Je nommerai k -motif commençant (k -motif^C) le k -motif en position p et k -motif finissant (k -motif^F) le k -motif en position $p + 1$ les motifs permettant de construire le $k + 1$ motif. Lorsque qu'une relation $r(p_i, p_{i+k})$ appartient à plusieurs pavés relationnels, l'instance p_i est ajoutée aux extensions de chaque motif possible (voir figure 7.13.B).

Il est possible d'augmenter le décalage des k -motifs relationnels pour que la croissance des motifs soit plus rapide. Dans ce cas, il faudrait vérifier, toutes les relations manquantes. Nous avons implanté une version de l'algorithme permettant de modifier le décalage, mais sans vérifier les relations manquantes (à part la relation entre la première et la dernière position). Cette version de l'algorithme est utile pour la recherche de motifs relationnels répétés dans les structures car, le squelette peptidique étant très contraint, il est inutile de vérifier toutes les distances. Il est possible de modifier le décalage de deux manières. La première est d'imposer une croissance fixe aux motifs, les motifs grandissent de l à chaque étape, le chevauchement des motifs est alors de longueur variable. Nous nommerons ce mode de croissance des motifs « décalage fixe ». La seconde est d'imposer un chevauchement de longueur fixe des motifs, la croissance des motifs est alors variable, comme dans l'algorithme de KMRC. Nous nommerons ce mode de croissance « chevauchement fixe ». Le premier cas présenté n'est en fait qu'un cas particulier du décalage fixe, le décalage étant de 1. Dans la section suivante, le décalage sera toujours de 1. Je reviendrai aux autres cas lors de la présentation de l'application aux structures.

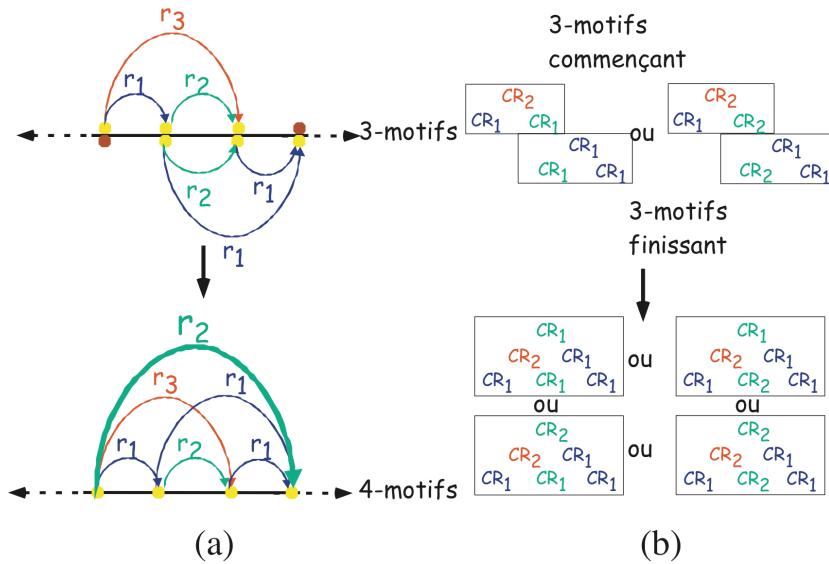


FIG. 7.13 – Construction de 4-motif relationnels à partir de 3-motifs relationnels chevauchants. Seule une instance de chaque motif est représentée mais on suppose que tous les motifs ont assez d'instances pour être répétés.

(a) Toutes les relations sont connues sauf la relation entre la première et la dernière position (en gras) qu'il faut vérifier.

(b) Comme la relation r_2 appartient à deux pavés relationnels, chaque instance de longueur 3 correspond à deux 3-motifs. Les 3-motifs sont combinés lorsque les pavés de relations se chevauchants sont les mêmes. Les premiers 3-motifs sont nommés « motifs commençant » (k -motif^C) et les seconds « motifs finissant » (k -motif^F). Deux 4-motifs sont donc construits. Cependant, lors de la vérification de la relation entre la première et la dernière position, deux pavés relationnels sont trouvés. Quatre 4-motifs sont donc créés.

Il faut remarquer que dans l'exemple, seuls sont combinés les 3-motifs relationnels se chevauchant sur le même pavé relationnel (pavé $\{CR_1, CR_2\}$). Cependant, ce cas est extrêmement simple car le chevauchement n'est fait que sur une position. Lorsque le chevauchement implique $k - 1$ positions, il faut que les $(k - 1)(k - 2)/2$ pavés relationnels soient identiques. Si nous nommons préfixe le k -motif sans le dernier symbole (et les relations de ce symbole) et suffixe le k -motif sans le premier symbole, cette condition peut être reformulée : il faut que le suffixe du premier k -motif soit identique au préfixe du second k -motif. Ces préfixes et suffixes sont des $k - 1$ -motifs relationnels, *i.e.* les motifs relationnels ayant servi à construire les k -motifs relationnels.

Il est important de ne générer un $k + 1$ -motif qu'à partir de deux k -motifs dont le suffixe du motif « commençant » est identique au préfixe du motif « finissant » car sinon, beaucoup de motifs inutiles sont générés. Nous nommons ces motifs **pseudo-motifs**. Il est prouvé dans l'article de N. Pisanti et coll. (Pisanti et al., 2005a) qu'ils sont inutiles pour construire tous les motifs maximaux suivants car tous les motifs générés par combinaison de pseudo-motifs sont non maximaux (*i.e.* ils sont tous inclus dans des motifs maximaux). Il a aussi été prouvé que les k -motifs maximaux suffisent pour construire tous les motifs maximaux de taille supérieure.

7.3.4 L'algorithme : implémentation avec des piles

Construire les motifs consiste, dans notre implémentation, à construire l'extension des motifs relationnels répétés maximaux.

1) Structures de données

L'algorithme que nous avons développé est similaire à celui de KMRC mais il contient en plus une « étape relationnelle », ainsi que l'utilisation des préfixes et suffixes. Les motifs et les extensions sont stockés dans des piles. Il y a alors des piles de motifs (les motifs sont représentés par un numéro d'index) et des piles de positions. Ces deux types de piles sont stockées à chaque étape dans deux vecteurs de piles :

V : vecteur de piles de taille égale à la longueur de la séquence. Il est donc indexé sur la séquence et il contient une pile de motifs à chaque indice ;

P : vecteur de piles indexé sur les numéros des motifs. Il contient à chaque indice i une pile de position, *i.e.* l'extension du motif i .

Deux autres vecteurs de piles sont construits temporairement au cours d'une étape de construction des motifs de taille +1. Chaque pile contient plusieurs extensions de motifs (*i.e.* plusieurs motifs). Pour séparer les extensions dans une même pile, un séparateur est inséré entre chaque extension. :

Qa : vecteur de piles construit lors de la première étape. Il a le même nombre de piles que p mais chaque pile contient plusieurs extensions à la suite (donc plusieurs motifs). Pour différencier les extensions qui sont dans une même pile, les extensions sont séparées par un chiffre négatif qui est le numéro du $k-motif^C$. Qa est indexé sur les numéros de $k-motif^F$ du suffixe. Dans une pile se trouvent donc tous les $k+1$ -motifs générés à partir des même $k-motif^F$, de différents $k-motifs^C$;

Qb : vecteur de piles construit lors de la seconde étape. Il a au maximum autant de piles qu'il y a de pavés relationnels ($|G_R|$). Le séparateur est la concaténation des numéros de $k-motif^C$ et de $k-motif^F$ présents dans les piles de Qa ;

2) Algorithme général de Triades

L'algorithme expliqué ici est illustré dans la figure 7.14. Les motifs de longueur $k+1$ sont construits à partir des k -motifs.

étape 1 Initialisation, construction du premier vecteur V (Init()).

Les pavés de symboles sont numérotés et à chaque position de V sont poussées tous les numéros de pavés auxquels appartient le symbole à cette position ;

étape 2 Construction du vecteur de piles Qa (BuildQa(), algorithme 4).

A partir du vecteur V est construit le vecteur P . Pour chaque pile de P , les éléments p_i (positions

dans la séquence) sont dépliés et chaque élément est poussé dans les piles de Q_a ayant pour indice les numéros de motifs trouvés dans V à l'indice $p_i + 1$. Dans l'exemple de la figure 7.14, la première position dans P (10) est la dernière de la séquence, elle est donc éliminée. Pour la seconde position (9), on regarde dans V en position +1 quels sont les numéros de pavés. Comme « 1 » est présent en position 10 de V , la position 9 est poussée dans la pile « _1 » de Q_a . De plus, comme on est en train de traiter le motif « 1 » de P , le numéro de k -motif « 1 » est associé à cette position dans Q_a . Ce « 1 » en gris dans Q_a est le numéro de préfixe du nouveau $k+1$ -motif. L'indice « _1 » est le numéro de suffixe. Lorsqu'une extension ne respecte pas le quorum, elle est éliminée (extension barrée dans Q_a) ;

étape 3 Construction du vecteur de piles Q_b (BuildQb(), algorithme 5).

Pour chaque pile de Q_a , les positions p_i sont dépliées. La relation entre p_i et $p_i + k$ est recherchée et p_i est poussé dans les piles de Q_b dont les indices correspondent aux pavés de relations auxquels appartient $r(p_i, p_i + k)$. Les numéros de préfixe et suffixe sont conservés pour chaque extension (numéros en gris dans Q_b). Dans l'exemple de la figure 7.14, la première position dans Q_a est 6. La relation $r(6, 7)$ appartient au pavé relationnel CR_1 (en bleu). La position 6 est donc poussée dans la pile de Q_b d'indice 1. Le quorum est vérifié (une extension est éliminée dans Q_b). Il est intéressant de remarquer que l'extension $\{9, 5, 1\}$ présente dans la première pile de Q_a n'existe plus dans Q_b car $r(9, 10)$ n'a pas de pavés commun avec $r(5, 6)$ et $r(1, 2)$: la relation entre les deux éléments est différente, les motifs relationnels sont donc différents. Cette extension est divisée en deux extensions : $\{5, 1\}$ dans la première pile de Q_b et $\{9\}$ dans la troisième pile. Il faut remarquer que comme les seules relations nécessaires sont celles entre les symboles i et $i + 2k - 1$, seules $n \times k$ relations sont mémorisées à chaque itération des étapes de construction ;

étape 4 Construction de la pile V (BuildV(), algorithme 6).

Cette étape consiste à re-numéroter les extensions (numéros oranges) et à pousser dans V les nouveaux numéros. Pour cela, toutes les positions sont dépliées jusqu'à ce qu'un séparateur soit rencontré. Un numéro est alors attribué à l'extension et ce numéro est poussé dans V à toutes les positions dépliées. Ces opérations sont effectuées jusqu'à ce que Q_b soit vide. Ainsi, dans l'exemple, l'extension $\{1, 2, 5, 6\}$ prend pour numéro 1 et 1 est poussé en $\{1, 2, 5 \text{ et } 6\}$ dans V ;

Etape 5 Filtrage de la pile V (FilterV(), algorithme 7).

Certaines des extensions créées ne sont pas maximales. Ces motifs inutiles sont éliminés lors de cette étape. Dans l'exemple, trois motifs sont éliminés. Lorsqu'une extension est incluse dans une autre, elle gagne les numéros de $motif^C$ et $motif^F$ de l'extension détruite. Ainsi, dans l'exemple, l'extension 2 (orange) est incluse dans l'extension 1 (orange). Elle donne son $motif^C$, 2 (gris), et son $motif^F$, 2 (gris), à 1 (orange). Ces numéros de $motif^C$ ou $motif^F$ sont les futures numéros de préfixes et suffixes qui seront utilisés à l'itération suivante. Le détail de l'algorithme de filtrage est dans l'algorithme 7. Les nouveaux motifs maximaux sont ensuite re-numérotés

(numéros roses) et stockés dans le vecteur V^{24} .

3) Complexité

La complexité dans le pire des cas de KMRC (sans relation) est $O(\log(k)ng_S^{2k})$ avec n la longueur du texte s (soit $|s|$), k la longueur des motifs trouvés et g_S la dégénérescence symbolique. Si les motifs de taille k sont construits à partir des motifs de taille $k/2$, ng_S^k motifs de longueurs k sont générés. En effet, en chaque position du texte il y a au plus $g_S^{\frac{k}{2}}$ $k/2$ -motifs, soit au total $ng_S^{\frac{k}{2}}$ $k/2$ -motifs. Chaque motif est concaténé avec au pire $g_S^{\frac{k}{2}}$ $k/2$ -motifs pour former les k -motifs. Ces k -motifs sont donc au nombre de ng_S^k . Lors de l'étape de filtrage, au pire, chaque extension d'un motif est comparée avec toutes les autres extensions ayant au moins une position commune, soit g_S^k extensions (motifs). Il y a donc au pire ng_S^{2k} comparaisons. Comme les étapes de sont répétées $\log(k)$ fois pour construire les motif de longueur k , la complexité dans le pire des cas de KMRC est en $O(\log(k)ng_S^{2k})$.

Lorsque les relations sont à prendre en compte la complexité dans le pire des cas de l'algorithme de Triades est en $O(kng_S^{2k}g_R^{2k^2})$. En effet, le nombre de motifs « symboliques » précédent est multiplié par le nombre de relations entre symboles d'un motifs. Il y a donc en une position du texte $ng_S^kg_R^{\frac{k(k-1)}{2}}$ k -motifs possibles. Lors de l'étape de filtrage, la complexité est donc en $O(n(g_S^kg_R^{k^2})^2)$ simplifiée en $O(ng_S^{2k}g_R^{2k^2})$. Comme les motifs croissent dans ce cas de 1 et non de k , les étapes sont répétées k fois, la complexité est donc en $O(kng_S^{2k}g_R^{2k^2})$.

7.3.5 Applications aux structures protéiques

Nous avons vu que pour appliquer cet algorithme aux structures, les positions représentent les C_α de la chaîne polypeptidique et les relations entre les positions sont les distances internes entre C_α échantillonnées suivant un pas ou maille (en Angstrom). L'espace des relations est ensuite découpé en pavés selon une marge (en unité de maille). Ainsi, avec une maille de 0,5Å et une marge de 2, la distance 1,5Å sera considérée comme similaire aux distances 0,5Å, 1Å, 2 et 2,5Å. Chaque pavé relationnel contient donc 5 relations et chaque relation appartient à 5 pavés relationnels (la dégénérescence est de 5). Comme seules les relations sont à prendre en compte dans les structures (et pas les symboles), il n'y a qu'un symbole dans l'alphabet non relationnel.

Avec une telle dégénérescence, il impossible de rechercher les motifs structuraux sur des structures réelles avec un décalage de 1 car trop de motifs sont générés. Un décalage est adéquat.

L'utilisation des préfixes et suffixes prend tout son sens lors de la comparaison de structures protéiques. En effet, comme la dégénérescence est très forte, chaque motif peut être dégénéré en beaucoup d'autres et beaucoup de pseudo-motifs pourraient être générés. Le tableau 7.1 présente le nombre de motifs répétés maximaux générés à chaque itération pour quatre structures (5^e colonne), ainsi que

²⁴Une autre structure des données est donc importante : à chaque extension est associée une liste de numéro de préfixes et de suffixes. Ces listes de préfixes et de suffixes sont stockées dans des arbres binaires de recherche (ABR) : il existe arbre pour chaque liste.

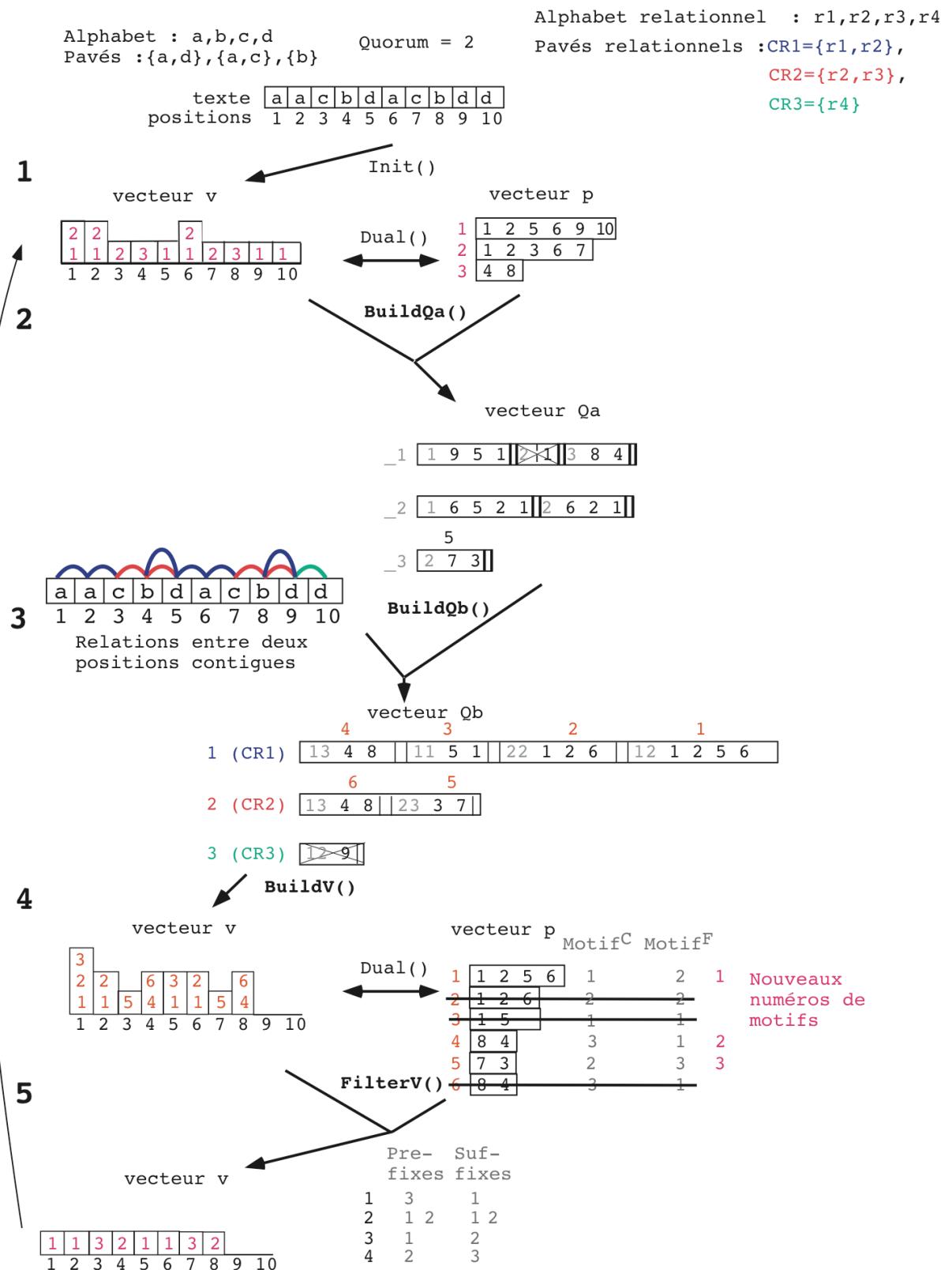


FIG. 7.14 – Exemple de fonctionnement de l'algorithme de Triades pour la séquence de symboles « a a c b d a c b d d ». Les pavés de symboles sont indiqués en haut à gauche. Les relations et pavés de relations sont définis en haut à droite. Les 2-motifs relationnels sont construits à partir des 1-motifs. Les relations entre positions ne sont données qu'entre les positions p et $p+1$, les seules nécessaires à cette étape. Comme il s'agit de la première construction de mots de taille $k+1$ à partir des mots de taille k ($k=1$), il n'y pas encore de préfixe ni de suffixe, mais les listes sont établies pour l'étape suivante de construction des 3 motifs. Pour une description détaillée, cf. Algorithme général de Triades page 142.

$\ell - 3 \rightarrow \ell$	occurrences générées	occurrences évitées	nombre de motifs	nombre de motifs maximaux	nombre d'occ. des motifs maximaux
$4 \rightarrow 7$	8 335	0	153	76	16 918
$7 \rightarrow 10$	104 006	77 882	7 565	1 439	84 390
$10 \rightarrow 13$	1 410 871	4 223 925	78 847	4 181	157 311
$13 \rightarrow 16$	4 438 142	23 143 653	97 401	628	4 896
$16 \rightarrow 19$	13 160	31 763	629	25	129
$19 \rightarrow 22$	41	43	5	2	9

TAB. 7.1 – Taille totale des extensions et nombre de motifs à différentes étapes et pour différentes longueurs de motif. Le nombre d'occurrences générées correspond à la taille totale des extension après l'étape 2 de construction de Q_a tandis que le nombre d'occurrences générées correspond au nombre de positions qui n'ont pas été mises dans Q_a car le suffixe et le préfixe différaient. La colonne suivante correspond au nombre de motifs générés à l'étape 2. Les deux dernières colonnes présentent le nombre de motifs maximaux répétés finaux ainsi que la taille total des extensions de ces motifs.

le nombre total de positions dans tous ces motifs (6^e colonne). La première colonne correspond au nombre de positions présentes dans Q_a à la fin de la première étape et la seconde correspond au nombre de positions qui n'ont pas été mises dans Q_a car les préfixes et suffixes différaient. Il est visible que ce nombre est très important. Le nombre important de motifs s'explique par le fait que les cytochromes P450 contiennent beaucoup d'hélices α assez longues, structures hautement répétitives mais avec cependant quelques variations. Ainsi une grande partie des motifs représentent des fragments d'hélice α un peu différents dont les extensions sont très chevauchantes. Par exemple, un motif d'extension $\{p_1, p_2, p_6, p_7, p_8\}$ et un motif d'extension $\{p_1, p_2, p_6, p_7, p_9\}$ sont très chevauchants. Ainsi, des 25 19-motifs, seuls cinq ne sont pas chevauchants.

7.3.6 Quelques exemples sur les cytochromes P450

Nous avons choisi quatre structures de cytochromes P450 : trois de bactéries (code PDB 1CPT, 2HPD chaîne B et 3CPP) et un de champignon (code PDB 1ROM). Il faut noter qu'ici l'algorithme recherche les motifs apparaissant dans les 4 structures. Les distances entre C_α sont discréétisées avec une maille de 1,5 Å, et la marge est de 1 (de sorte que la dégénérescence soit 3). Le décalage est fixe et il est de 3. Les résultats sont les cinq motifs qui sont montrés dans Figure 7.15. De tels motifs ont été précédemment identifiés par P. Jean et coll. (Jean et al., 1997), en utilisant différentes techniques, sur 3 de ces protéines. Ceci montre la sensibilité de la méthode qui est présentée ici. La figure 7.16 illustre les alignements des fragments des structures correspondant aux différents motifs. Ceux-ci prouvent que cette méthode est capable de trouver les structures localement conservées, et par conséquent que les relations sont un bon choix pour représenter l'information structurale.

7.3.7 Conclusions

Les avantages de cette méthode est qu'elle est réellement multiple, exhaustive (tous les motifs maximaux sont construits), et générique. Dans le cas des structures, il faut cependant accepter que

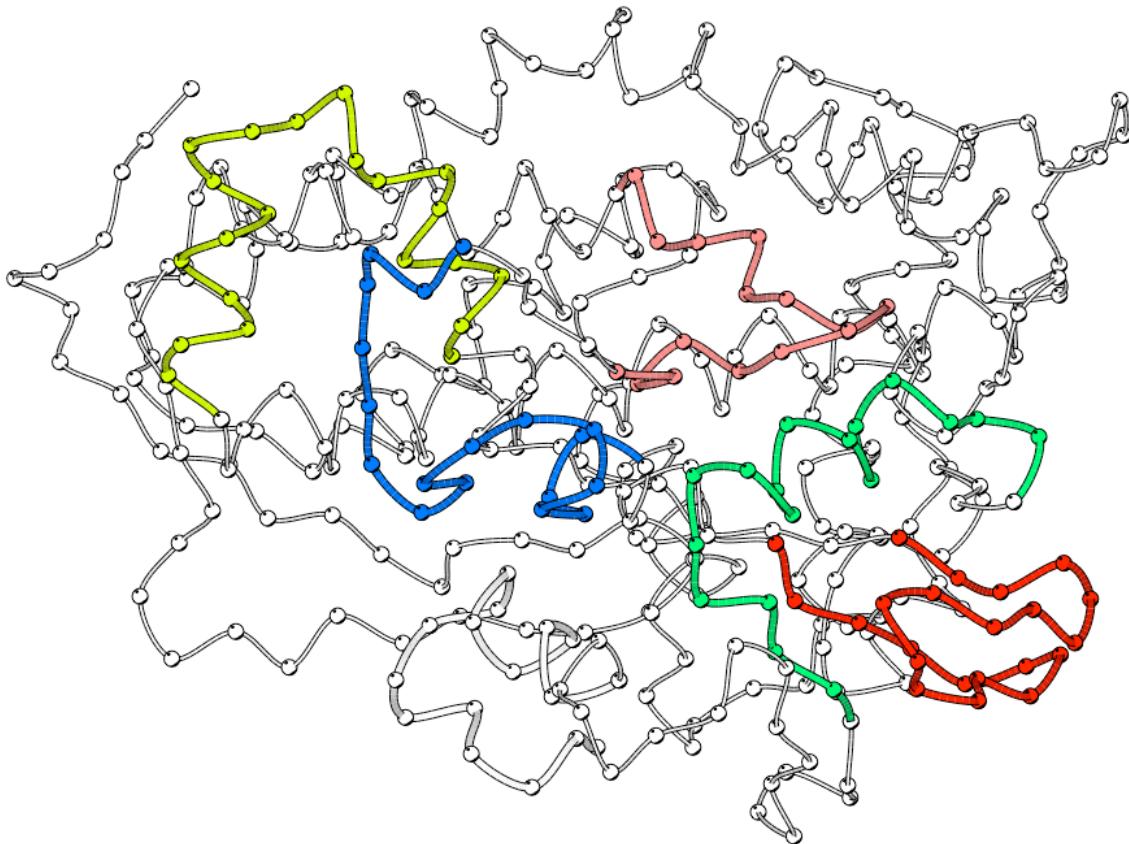


FIG. 7.15 – Cinq motifs structuraux trouvés dans quatre cytochromes P450 : deux de 22 résidus, deux de 19 et un de 16. Seule une structure est affichée (3CPP).

certaines distances ne soient pas vérifiées (la dégénérescence est trop grande), néanmoins les motifs structuraux trouvés sont tout à fait similaires. Les contraintes géométriques de l'enchaînement des distances rend ces vérifications en fait inutiles. La principale limite de la méthode réside dans le mode de construction des motifs et l'exhaustivité. Lorsque le nombre total de résidus n dans toutes les structures augmente, le nombre de fragments de structures candidats augmente ce qui peut poser des problèmes de mémoire (beaucoup de motifs non maximaux sont générés lors des étapes intermédiaires)

De plus, quelques traitements finaux seront nécessaires pour que les motifs structuraux trouvés forment un alignement structural multiple. Pour l'instant, tous les motifs maximaux d'une taille donnée et de taille inférieure sont fournis par la méthode. Pour avoir un alignement multiple de structures, il faudra trouver la meilleure combinaison de motifs de tailles variables (opération pour l'instant possible uniquement sur les motifs de même taille). Il sera aussi possible et intéressant d'avoir des motifs

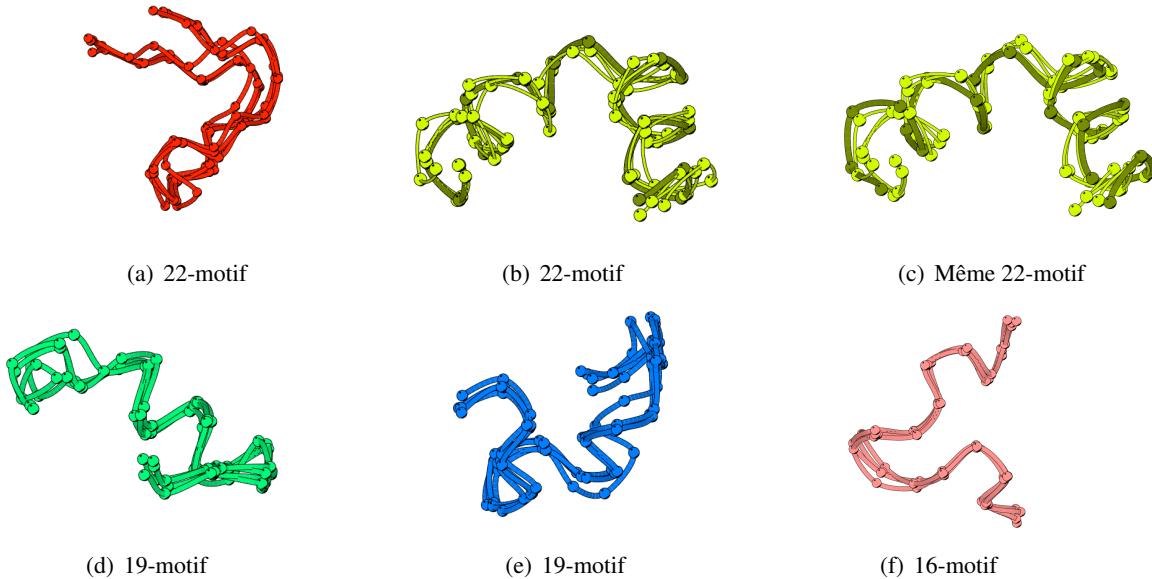


FIG. 7.16 – Détail des cinq motifs structuraux trouvés dans les 4 cytochromes P450. Le second motif est montré deux fois (7.16(b) and 7.16(c)) car il a deux occurrences qui se chevauchent dans la protéine 1 (le premier débute en position 248 et le second au résidu 258). Ces occurrences sont de couleur plus foncée.

de taille variable suivant la protéine. En effet, lors de la construction des motifs de taille k à partir de motifs de taille $k - 1$, il est possible qu'une protéine présente dans l'extension du $k - 1$ -motif ne le soit plus dans l'extension du k -motif. Connaissant les motifs de taille k et ceux de taille $k - 1$ ayant permis de les construire, il est possible de construire un bloc structural avec les motifs de taille k pour certaines protéines et les motifs de taille inférieure lorsque c'est possible.

Troisième partie

Résultats

CHAPITRE 8

Constitution des familles structurales

Plusieurs classifications des structures protéiques sont disponibles à l'heure actuelle. SCOP, CATH, MMDB, FSSP et CE database en sont les plus connues. Ces classifications sont obtenues par des approches totalement manuelles, partiellement manuelles ou totalement automatiques. Néanmoins aucune de ces classifications n'est basée sur une comparaison locale des structures protéiques. Pourtant, dans certains cas, une comparaison locale des structures est plus adéquate, par exemple pour définir les cœurs structuraux qui peuvent ensuite être utilisés pour la reconnaissance de repliements. (*threading*) mais aussi parce que les structures des protéines sont modulaires. Ainsi, deux protéines pourtant globalement très différentes peuvent avoir des modules ou domaines structuraux très similaires. Ce problème est résolu dans CATH et SCOP par une subdivision des structures protéines en domaines structuraux mais cette procédure est loin d'être triviale et les résultats sont quelque fois assez différents.

Nous avons mis en place une procédure permettant une classification non hiérarchique des structures protéiques où seules les ressemblances locales sont prises en compte. Ensuite, les protéines appartenant à une même famille sont comparées structuralement pour obtenir le cœur de cette famille, composé de blocs structuraux similaires. Les résultats préliminaires de cette procédure seront présentés. Pour que les cœurs soient utilisables pour la reconnaissance de repliements. il faudrait en outre une vérification plus poussée des familles constituées et une meilleure adaptation de la procédure de définition des cœurs.

Aucune des classifications à ce jour accessibles ne répond à certains critères qui me semblent importants. Tout d'abord, il faudrait que la procédure de classification soit totalement automatique et basée sur des critères précis. Ensuite, une classification hiérarchique de toutes les familles de structures n'est sans doute par l'approche la plus adéquat pour analyser cette question. En effet, la classification est alors effectuée sont souvent basées sur le contenu en structures secondaires des protéines, mais il peut n'y avoir aucune autre relation entre les structures. Le troisième critère est que la classification structurale devrait être faite indépendamment de l'information de séquence²⁵. Enfin, puisque le découpage d'une structure en domaines est un problème épineux, il est particulièrement approprié d'assigner une structure à plusieurs familles.

La procédure de classification se déroule en trois étapes : i) comparaison de toutes les structures deux à deux, ii) classification en paires de structures similaires et non similaires, iii) regroupement en familles des structures similaires (voir figure 8.1).

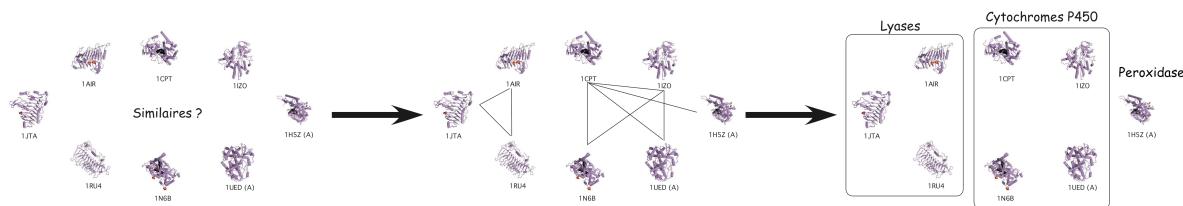


FIG. 8.1 – Procédure de classification des structures en familles. Toutes les paires de structures sont comparées. Les structures sont ensuite regroupées en familles selon les similarités trouvées.

La banque de structures²⁶ protéiques choisie est une banque réalisée localement qui contient des structures ayant moins de 90% d'identité de séquence entre elles (15056 structures).

Première étape La première étape ne pose aucune difficulté particulière car elle consiste à comparer toutes les structures entre elles avec le programme YAKUSA. Le temps de recherche sur la banque avec une structure requête étant d'environ 80 secondes, il faut environ 2 semaines sur un Macintosh G5 1.2 Ghz pour effectuer toutes les comparaisons.

Seconde étape L'objectif de la seconde étape est d'identifier parmi toutes les paires de structures celles qui sont composées de structures similaires. Lorsqu'une structure requête est comparée à une banque par YAKUSA, les structures de la banques sont classées selon le Z-score (*cf.* page 111). Ce Z-score est un bon indicateur de similarité entre deux structures. Les paires de structures de Z-score inférieur à 4 sont la plupart du temps non similaires. Cependant, pour les paires de Z-score supérieur

²⁵L'information de séquence ne doit bien entendu pas être totalement oubliée, mais il me semble judicieux de n'utiliser cette information qu'après la procédure de classification.

²⁶La structure de certaines protéines est résolue sous forme de complexe, ces structures « PDB » sont alors composées de plusieurs « chaînes », rassemblées en un seul fichier dans la PDB. Cependant, lorsque le terme de « structure » sera ici utilisé, il correspondra toujours à une seule chaîne, les structures PDB ayant plusieurs chaînes étant divisées en autant de fichiers qu'elles comportent de chaînes.

à 4, le Z-score ne suffit pas toujours pour identifier les paires de structures similaires. Dans le meilleur des cas, toutes les structures similaires à la structure requête sont classées en premier et la valeur du Z-score entre la dernière structure similaire à la requête et la structure suivante dans le classement chute assez fortement. Dans les autres cas, la chute peut ne pas être très prononcée (par exemple si la requête est très riche en hélices α) et des structures non similaires à la requête peuvent être alors classées avant des structures similaires. Le Z-score ne suffit donc pas à identifier les paires de structures similaires. Il faut alors prendre en compte d'autres critères de classification.

Lors de l'évaluation d'un alignement structural par inspection visuelle, plusieurs critères entrent en compte. Il est difficile de connaître exactement tous ces critères, car ils peuvent varier d'une personne à l'autre et sont assez nombreux. Ils peuvent être : la longueur de l'alignement, la compatibilité spatiale des blocs structuraux, le $RMSD_c$ de chaque bloc... J'ai donc essayé de prendre en compte tous ces critères pour déterminer automatiquement si deux structures sont similaires ou non similaires (*i.e.* ont des ressemblances dues au hasard). Ceci consiste à classer les paires de protéines en deux classes : paires de structures similaires et paires de structures non similaires. Deux méthodes de classifications ont été testées : les Machines à vecteurs de support (SVM pour *Support Vector Machines* ou) et les lois Normales Multivariées (MVND pour *MultiVariate Normal Distributions*).

Troisième étape L'objectif est maintenant de regrouper les structures similaires en familles de structures, le nombre de familles étant inconnu. La méthode de classification utilisée doit alors : ne pas voir un nombre fixe de classes et ne pas avoir besoin de distance²⁷ entre les structures. Les méthodes de partitionnement de graphes répondent à ces critères. La méthode MCL (van Dongen, 2000; Enright et al., 2002) a été utilisée.

8.1 Classification des paires de structures

L'objectif est de classer les paires de structures en deux classes (paires de structures similaires et paires de structures non similaires) à partir des informations fournies par YAKUSA. Je présenterai d'abord brièvement les deux méthodes utilisées, le jeu de données d'apprentissage et enfin les résultats obtenus. Je nommerai « éléments à classer » les paires de protéines (ou paires) à classer et « critères » les informations prises en compte. Ces critères forment un vecteur pour chaque paire de structures comparées.

8.1.1 Présentation des deux méthodes de classification

1) Les SVM

La méthode des SVM consiste à séparer deux ensembles de points par un hyperplan. L'idée originale des SVM a été publiée par Vladimir Vapnik (Boser et al., 1992). Elle est basée sur l'utilisation de

²⁷Une distance est une mesure réflexive, symétrique et avec laquelle l'inégalité triangulaire est vérifiée

noyaux (kernels) qui permettent une séparation optimale (sans problème d'optimum local) des points du plan en différentes catégories²⁸.

Par exemple, dans un espace à deux dimensions, sont répartis deux groupes de points : les point (+) pour $y > 0$ et les points (-) pour $y < 0$. Un séparateur linéaire évident existe entre ces deux ensembles, l'axe des abscisses. Le problème est dit linéairement séparable.

Pour des problèmes plus complexes, la caractérisation d'un séparateur linéaire peut être très compliquée et tout à fait non optimale (voir figure 8.2). Afin de remédier au problème de l'absence de séparateur linéaire, l'idée des SVM est de reconsidérer le problème dans un espace de dimension supérieure. Dans ce nouvel espace, il existe un séparateur linéaire qui permet de classer au mieux les points dans les deux ou plus groupes qui conviennent. Le séparateur linéaire obtenu est un hyperplan, c'est à dire la généralisation à n dimensions d'une ligne (1D) séparant un espace 2D, ou d'un plan (2D) séparant un espace 3D. La méthode fait appel à un jeu de données d'apprentissage, qui permet d'établir un hyperplan séparant au mieux les points. Plus la dimension de l'espace de description est grande, plus la probabilité de pouvoir trouver un hyperplan séparateur entre les points est élevée. Le calcul de l'hyperplan dans un espace de très grande dimension est en général effectué grâce à une fonction noyau par exemple de type polynomiale.

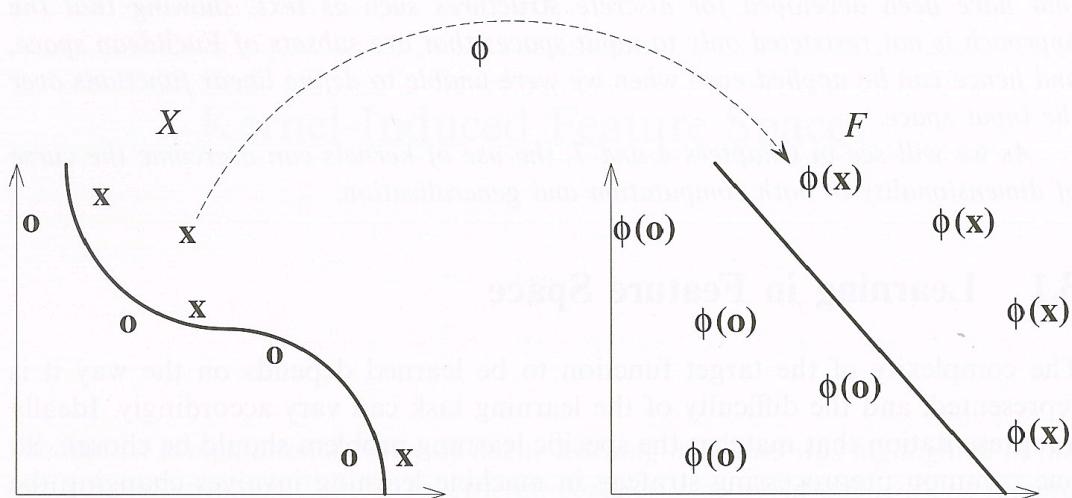


FIG. 8.2 – Figure extraite du livre de N. Cristianini and J. Shawe-Taylor (Cristianini and Shawe-Taylor, 2000). La transformation ϕ de l'espace d'entrée (à gauche) en espace de re-description permet de trouver un séparateur linéaire des deux ensembles de points. Ici, les deux espaces ont deux dimensions mais en général, l'espace de redescription est de dimension supérieure à l'espace de départ.

J'ai utilisé une implémentation des SVM nommée SVM Torch (Collobert and Bengio, 2001), qui peut être téléchargée à l'adresse http://www.idiap.ch/machine_learning.php?content=Torch/en_SVMTorch.txt. La fonction noyau par défaut est une fonction à base radiale de la forme $K(a,b) = \exp(-|a-b|^2/(std^2))$ avec std une constante à définir par l'utilisateur (10 par défaut). Les

²⁸Pour plus d'informations : http://fr.wikipedia.org/wiki/Support_vector_machine

autres fonctions noyaux implémentées sont linéaires, polynomiales (de la forme $K(a, b) = (s \times a \times b + r)^d$, s , r et d étant définis par l'utilisateur) et sigmoïdes de la forme $K(a, b) = \tanh(s \times a \times b + r)$, s et r étant définis par l'utilisateur.

2) Les Lois Normale Multivariées

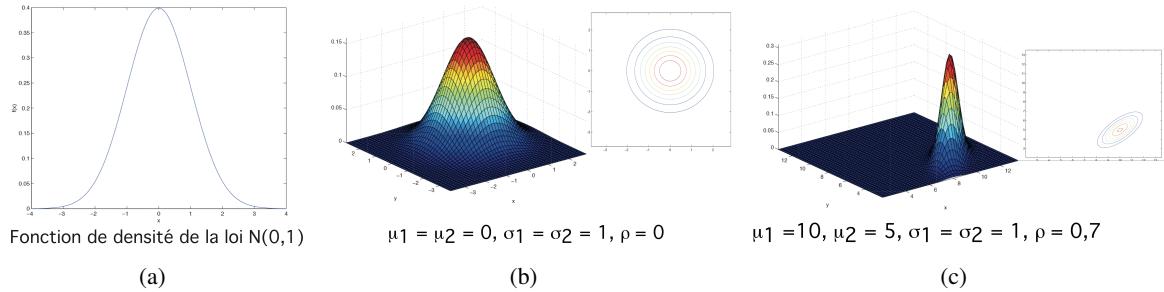


FIG. 8.3 – Exemples de fonctions de densités de lois Normales monovariées (a), bivariées (b et c).

La loi Normale monovariée de la variable aléatoire x d'écart type σ et de moyenne μ est définie par la densité de probabilité

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2} \quad (8.1)$$

La loi Normale bivariée des variables aléatoires (x_1, x_2) d'écarts types σ_1, σ_2 de moyennes μ_1, μ_2 et de covariance ρ est définie par la densité de probabilité (voir aussi figure 8.3) :

$$f(x_1, x_2) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} e^{-\frac{1}{2(1-\rho^2)}\left((\frac{x_1-\mu_1}{\sigma_1})^2 + (\frac{x_2-\mu_2}{\sigma_2})^2 - 2\rho\frac{(x_1-\mu_1)(x_2-\mu_2)}{\sigma_1\sigma_2}\right)} \quad (8.2)$$

Ainsi, si les deux variables sont indépendantes ($\rho = 0$), la fonction de densité se simplifie et devient le produit de deux fonctions de densités monovariées.

Une loi Normale multivariée de d variables aléatoires $X = (x_1, \dots, x_d)$, notée $\mathcal{N}(\mu, \Sigma)$ avec μ vecteur des d moyennes et Σ matrice des variances/covariances a pour densité de probabilité :

$$f(x_1, \dots, x_d) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(X-\mu)' \Sigma^{-1} (X-\mu)} \quad (8.3)$$

$|\Sigma|$ est le déterminant de la matrice Σ , $(X-\mu)' \Sigma^{-1} (X-\mu)$ est la distance de Mahalanobis de X à la loi Normale.

Dans notre cas, deux distributions Normales multivariées sont calculées, une pour chaque classe. La distribution de chaque caractéristique est donc approximée par une loi Normale même si elle n'est pas Normale. Un jeu de données test est donc nécessaire pour calculer la matrice de variance et les moyennes des deux distributions. Pour savoir si une paire de protéine appartient à l'une ou l'autre classe, la densité de probabilité pour chaque distribution est calculée. Les deux probabilités

sont ensuite normalisées ($p_1 \text{ normalize} = p_1 / (p_1 + p_2)$). Ces probabilités normalisées ont une valeur comprise entre 0 et 1 et leur somme est égale à 1. La paire de protéine dont la classe doit être prédite est attribuée à la classe similaire si la probabilité normalisée d'appartenir à cette classe est supérieure à un seuil donné, par exemple 0,5. Ce seuil est le seul paramètre à régler avec cette méthode (que je nommerai MVND). Elle a été implémentée par A. Marin pour le typage en acide aminé des systèmes de spins observés en RMN des proéines. (Marin et al., 2004).

8.1.2 Jeu de données d'apprentissage

Un jeu de données d'apprentissage où la classe est définie pour les paires de structures est nécessaire à ces deux méthodes. Pour mettre en place un tel jeu de données, des structures représentatives des structures cristallisées à ce jour ont été comparées aux structures de la banque (PDB non redondante). L'objectif est d'identifier les paires de structures similaires parmi les meilleurs résultats de YAKUSA. Il est inutile de rechercher des structures similaires dans les résultats ayant un Z-score trop bas. Ainsi, seuls les 50 meilleurs résultats ont été conservés pour chaque structure requête. Les structures requêtes choisies sont les protéines représentatives de chaque classe *fold* (second niveau) de SCOP (772 protéines) mais la banque est la banque non redondante.

Pour savoir si les structures d'une paire sont similaires ou non, les données des trois classifications SCOP, CATH et FSSP ont été croisées. Les règles d'identification des paires de structures similaires et non similaires sont :

- Si les deux protéines sont dans les mêmes classes de niveau 3 de SCOP et CATH (respectivement classe *superfamily* et classe *Topology*), les deux protéines sont considérées comme similaires.
- Si les deux protéines sont dans deux classes de niveau 2 différentes dans SCOP et CATH (respectivement classe *fold* et classe *Architecture*), elles sont considérées comme différentes.
- Si les deux protéines sont dans l'une des deux classifications dans la même classe de niveau 3 et dans l'autre dans la même classe de niveau 2 ou absente, la classification de ces deux protéines est vérifiée dans FSSP. Si elles sont dans la même famille FSSP, les deux protéines sont considérées comme similaires. Et inversement pour les structures différentes.
- dans tous les autres cas, les paires ne sont pas retenues pour le jeu de données car il semble difficile de les attribuer à une classe.

Ces conditions semblent assez strictes pour avoir un jeu de données fiables. Sur les 38 600 paires de structures (772×50), 3 920 sont similaires selon les 3 classifications, 1 120 sont similaires dans SCOP et CATH et absentes dans FSSP. De même, pour les paires de structures non similaires, 8 403 sont dans des classes de niveau 2 différentes dans SCOP et CATH et dans des familles différentes dans FSSP et 12 067 sont différentes selon SCOP et CATH mais absentes dans FSSP. Comme beaucoup de structures sont manquantes dans FSSP, seules 145 paires sont similaires et 1 891 d'après la troisième condition.

Nous avons donc au total 5 186 paires de structures similaires et 22 361 paires de structures diffé-

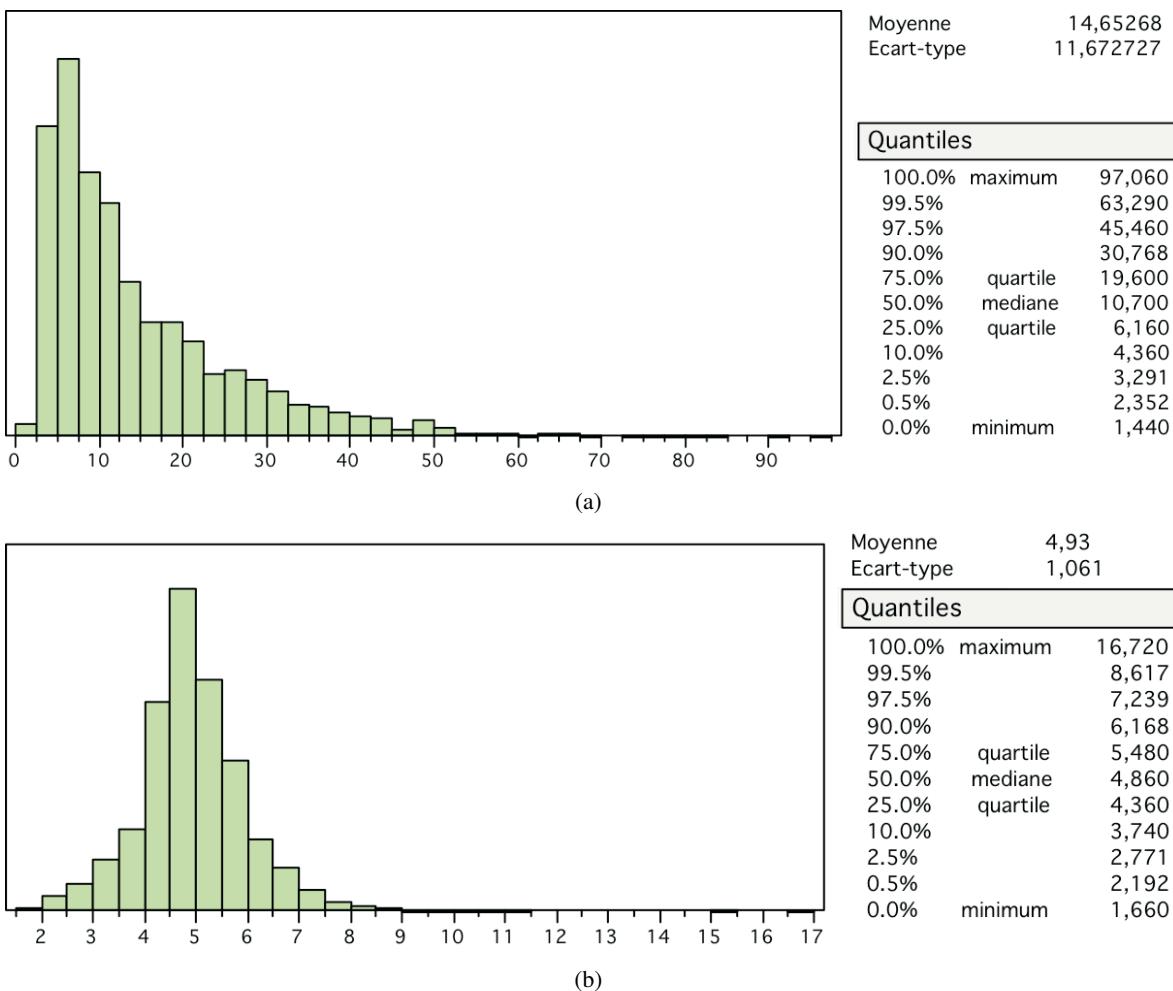


FIG. 8.4 – Distribution des Z-score de YAKUSA pour les paires de protéines similaires (haut) et différentes mais qui sont dans les 50 meilleurs résultats de YAKUSA (bas).

rentes. Une vérification manuelle a ensuite été effectuée pour les protéines similaires ayant Z-scores très faibles ou les protéines différentes ayant un Z-score très haut. Comme il est attendu, peu de protéines similaires ont un Z-score inférieur à 4 (voir figure 8.4) tandis que peu de protéines ayant un Z-score supérieur à 6 sont différentes.

En théorie, comme pour les SVM, les résultats de classification sont meilleurs lorsque les exemples des deux classes sont en nombres égaux, le nombre adéquat de paires de structures est tiré au hasard parmi toutes les paires différentes. Le jeu de test contient au final autant paires de protéines similaires que de paires de protéines non similaires.

8.1.3 Résultats des classifications

Il faut maintenant choisir les critères selon lesquels les structures seront classées.

1) Comparaison des résultats des SVM et des MVND

Le premier vecteur de critères est le vecteur 1 du tableau 8.1. Les résultats des SVM et des MVND ont été comparés uniquement pour ce vecteur. Il contient la plupart des données fournies par YAKUSA à propos d'un l'alignement structural : la longueur des deux protéines, le nombre de SHSP compatibles spatialement ainsi que la somme de leur longueur (la longueur de l'alignement), la moyenne de leurs $RMSD_c$ croisés, la somme de leurs scores angulaires et enfin le Z-score.

Pour tester l'efficacité des deux méthodes de classement, l'apprentissage a été réalisé avec tous les vecteurs du jeu de données sauf un. La classe du vecteur manquant est ensuite prédite. Cette opération est répétée pour chaque vecteur du jeu de données, les nombres de bonnes et de mauvaises prédictions pour chacune des deux classes peuvent alors être calculés. Pour simplifier les notations, les paires de structures appartenant à la classe des paires de structures similaires seront nommées éléments positifs et celles appartenant à la classe des paires de structures différentes les éléments négatifs. Ainsi, la sensibilité est le nombre de positifs correctement prédits ou vrais positifs rapportés au nombre total de positifs. La spécificité est le nombre de négatifs correctement prédits rapporté au nombre total de négatifs. La spécificité est donc la sensibilité pour la classe des protéines non similaires. La valeur prédictive positive est le nombre de positifs correctement prédits rapporté au nombre total de positifs prédits et la valeur prédictive négative est le nombre de négatifs correctement prédits rapporté au nombre total de négatifs prédits. Ces quatre valeurs, pour les prédictions effectuées par les SVM et les MVND, sont présentées dans le tableau 8.2.

La classification par les MVND est plutôt meilleure pour tous les seuils. Elle est cependant moins sensible pour la détection des paires de structures similaires sauf si le seuil est très bas. Ce seuil peut être très bas et pourtant permettre une bonne classification car la probabilité (non normalisée) d'appartenir à la classe des paires similaires a des valeurs globalement plus faibles que celles d'appartenir à la classe des paires différentes. Ceci provient probablement du fait que la distribution des critères de la classe des paires similaires est plus étroite que celle de la classe des paires différentes. De plus, les protéines très similaires n'ont pas une très forte probabilité d'appartenir à la distribution des paires de protéines similaires. En effet, certains des critères sont orientés, par exemple, plus les protéines se ressemblent, plus le Z-score est haut. Dans ces cas, une fonction cumulative est plus appropriée. Comme la probabilité d'appartenir à la distribution des paires de protéines non similaires de ces paires de protéines est nulle ou pratiquement nulle, elles sont cependant correctement classées.

Comme les résultats avec les MVND sont les meilleurs obtenus, que cette méthode est plus rapide lors de la phase d'apprentissage et qu'elle est facile à mettre en place, elle a été retenue malgré quelques légers défauts qu'il faudra corriger. La rapidité a été un critère de sélection de cette méthode car cela a permis de tester plusieurs autres types de vecteur de critères. Cependant, pour avoir des tests fiables pour les SVM, il faudrait faire varier les paramètres de la fonction noyau et surtout essayer d'autres fonctions noyaux.

Plusieurs types de vecteurs de critères ont ensuite été testés avec les MVND. Ce sont les vecteurs 2 à 5 du le tableau 8.1. Seuls deux critères ont été ajoutés : la moyennes des $RMSD_c$ croisés rapportées

Description	vecteur 1	vecteur 3	vecteur 2	vecteur 4	vecteur 5
Nombre de résidus de la protéine requête	X				
Nombre de résidus de la protéine banque	X				
Somme des longueurs des SHPSs compatibles spatialement formant le chemin le plus long (ou longueur de l'alignement)	X	X	X		
Nombre de SHPS compatibles spatialement formant le chemin le plus long	X	X	X		
Moyenne des $RMSD_c$ croisés des SHPS compatibles spatialement formant le chemin le plus long. Cette moyenne ne contient donc pas les $RMSD_c$ des SHSP eux mêmes, seulement les RMSs croisés). Pour les chemins avec 1 seul SHSP, cette valeur est mise à la moyenne (4,62).	X				
moyenne de $RMSD_c$ (comme ci-dessus) / nombre de SHSP compatibles spatialement formant le chemin le plus long		X	X	X	X
Somme des scores angulaires des SHPS compatibles spatialement formant le chemin le plus long	X	X		X	
Somme des scores angulaires / longueur totale de l'alignement			X		X
Z-score	X	X	X	X	X
% de résidus alignés par rapport à la structure de la banque		X	X	X	X
% de résidus alignés par rapport à la structure requête	X	X	X	X	X

TAB. 8.1 – Les 5 vecteurs de critères testés

	Sp	VPN	Se	VPP
MVND, seuil 0,5	97,65	85,02	82,60	97,24
MVND, seuil 0,45	97,32	85,41	83,37	96,89
MVND, seuil 0,35	96,57	86,23	84,57	96,11
MVND, seuil 0,1	91,57	90,36	90,22	91,46
SVMTorch	89,53	86,46	85,97	89,14

TAB. 8.2 – Sensibilité (Se) et spécificité (Sp), valeur prédictive positive (VPP) et valeur prédictive négative (VPN) des SVM et MVND obtenues avec le vecteur 1.

au nombre de SHSP, car ces RMSD dépendent très fortement du nombre de SHSP, et la somme des scores angulaires rapportée à la longueur de l'alignement, car elle y est directement liée.

Les sensibilités et spécificités pour la classe de paires de chaque type de vecteur sont présentées figure 8.5. Une courbe ROC (*Receiver Operating Characteristic*) a aussi été tracée pour la sensibilité et la valeur prédictive positive (voir figure 8.6). Cette courbe ne représente pas la sensibilité vs. la spécificité comme c'est habituellement le cas. Pour ensuite regrouper les protéines en familles, il semble plus important de ne pas prédire comme similaires des paires de structures qui ne le sont pas. Le risque est alors de ne pas prédire comme similaires des protéines qui le sont. En effet, *a priori*, ce risque pourra engendrer trop de familles, certaines familles pourraient être divisées car certains liens de similarités manquent.

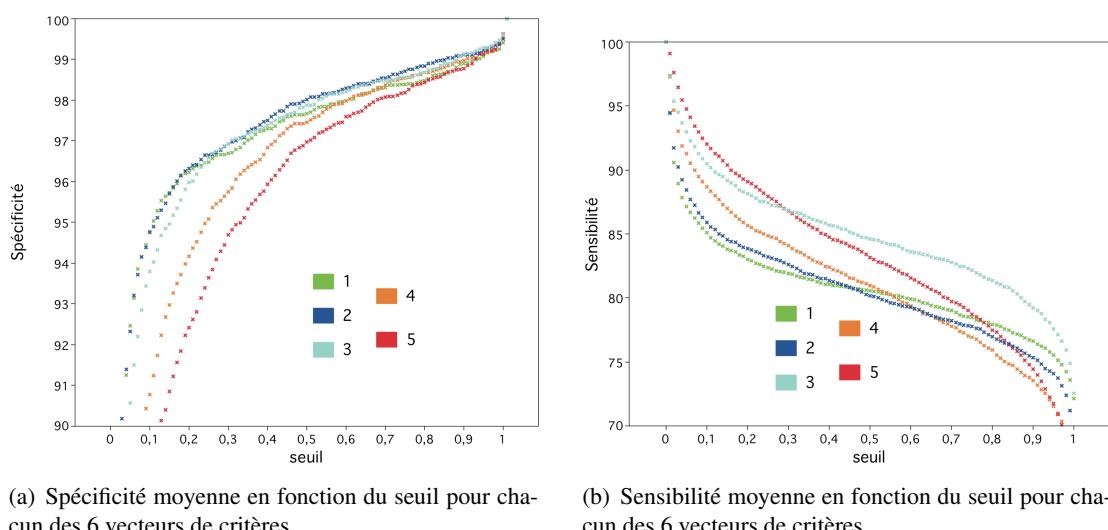


FIG. 8.5 – Moyenne des spécificités et sensibilités de deux classes. Chaque couleur correspond à un type de vecteur. Le vecteur 3 donne de bien meilleurs résultats que tous les autres.

Lorsque la moyenne des $RMSD_c$ croisés du vecteur 1 et remplacée par la moyenne divisée par le nombre de SHSP (vecteur 2, bleu foncé), la sensibilité et la spécificité sont légèrement améliorées. Lorsque la somme des scores angulaires est remplacée par la somme rapportée à la longueur de l'alignement (vecteur 3, bleu clair), les sensibilités sont supérieures à celles de tous les autres vecteurs.

La courbe ROC 8.6 indique que le vecteur 3 est celui permettant d'avoir les meilleurs résultats quant à la classe des paires de structures similaires. Les critères « nombre de SHSP » et « longueur de l'alignement » étant alors répétés, ces informations ont été ôtées des vecteurs (vecteurs 4 et 5, en vert). Les résultats sont alors moins bons, le type de vecteur 3 a donc été sélectionné.

Beaucoup d'autres types de vecteurs pourraient être testés, notamment en ajoutant des informations sur les SHSP non compatibles spatialement avec le groupe de SHSP formant le plus long alignement (celui pris en compte pour l'instant). En effet, ceux-ci, s'ils sont compatibles entre eux, peuvent signifier qu'un second domaine structural est similaire dans les deux structures. J'ai aussi essayé de calculer un $RMSD_c$ global en superposant tous les résidus alignés (dans les SHSP compatibles spa-

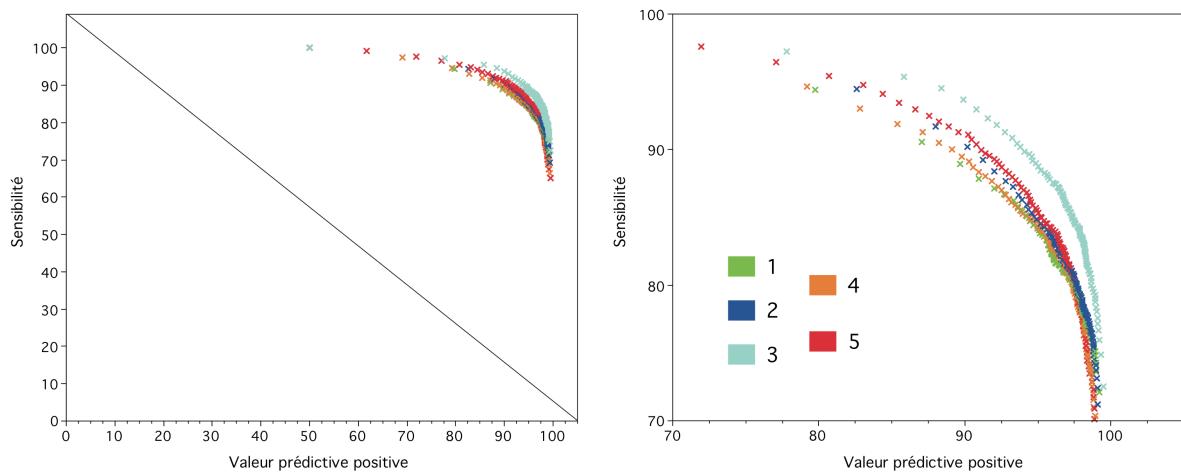


FIG. 8.6 – Courbe ROC sensibilité vs. valeur prédictive positive pour les 5 types de vecteur testés. Le vecteur 3 (bleu clair) est celui dont la surface sous la courbe est la plus grande, et donc celui permettant d’obtenir les meilleures prédictions pour la classe des paires de protéines similaires. Le seuil permettant d’obtenir à la fois la meilleure sensibilité et la meilleure valeur prédictive positive est celui pour lequel la distance à la diagonale est maximale. Ce seuil est d’environ 0,25.

tialement). La fonction utilisée (celle écrite par M. Sippl) ne converge pas dans la plupart des cas probablement parce que les C_α sont trop dispersés. Il n'est donc pas possible généralement de calculer la transformation permettant de minimiser le $RMSD_c$ global avec cette fonction (il faudrait peut-être choisir une autre procédure de superposition).

L'objectif de cette classification est de pouvoir déterminer automatiquement quelles sont les paires de protéines similaires parmi toutes les paires de structures comparées. Le regroupement des protéines en familles structurales sera fait sur la base de ces similarités. J'ai choisi, en fonction de cet objectif, un seuil égale à 0,25, pour lequel la spécificité est de 96,6% et la sensibilité de 87,4%, la valeur prédictive positive de 96,2% et la valeur prédictive négative de 88,5%.

2) Conclusion

Pour ce problème de classification, la méthode utilisant les lois Normales multivariées semble à première vue donner de meilleurs résultats que la classification avec les SVM. Cependant, des tests plus poussés sur l'utilisation des SVM et l'utilisation de fonctions cumulatives multivariées pourraient encore améliorer ces résultats.

Les probabilités calculées par MVND ne sont pas dans notre cas une mesure de similarité entre structures à cause du fait que certains des critères sont orientés. Ces probabilités n'ont donc pas été prises en compte lors de l'étape suivante de classification en familles des structures que nous allons maintenant aborder.

8.2 Classification en familles des structures

L'objectif est maintenant de construire les familles de structures protéiques similaires en structures. Comme le nombre de familles n'est pas connu et qu'il n'y a pas de distance définie entre les structures, j'ai choisi d'utiliser une méthode permettant de trouver les sous-graphes fortement connexes d'un graphe. Chaque structure est alors un sommet du graphe et une arête est définie entre deux sommets si elles ont été classées comme similaires à l'étape précédente. La méthode choisie est MCL (*Markov Cluster Algorithm*) (van Dongen, 2000; Enright et al., 2002).

8.2.1 Présentation de MCL

Cette méthode de partitionnement de graphe a déjà été appliquée à la détection de familles de protéines à partir de l'information de séquence fournie par PSI-BLAST (Enright et al., 2002). Son principe est que les groupes (*clusters*) de protéines similaires partagent beaucoup d'arêtes et donc que des marches aléatoires partant d'un sommet de ce groupe aboutiront le plus fréquemment à un autre sommet du même groupe. Elle peut être appliquée aux graphes non orientés, pondérés ou non.

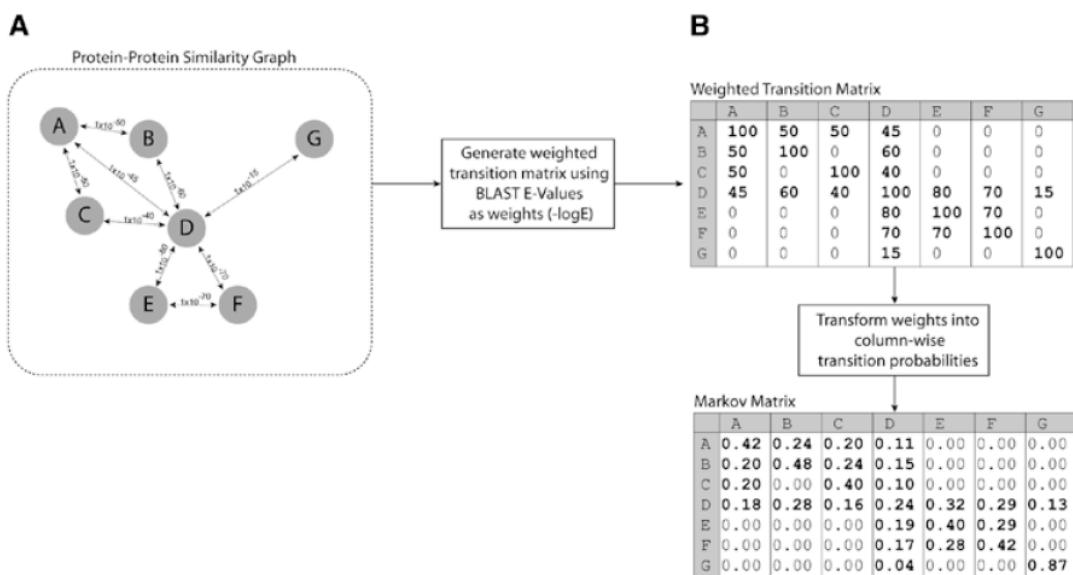


FIG. 8.7 – Figure extraite de l'article de Enright et coll. (Enright et al., 2002). (A) Exemple d'un graphe construit à partir des similarités de séquence de 7 protéines (A à F). Les ronds représentent les structures et les flèches les similarités trouvées par BLAST (sous la forme de *E-value*).

(B) Matrice de transitions pondérée (la pondération est de 0 si aucune similarité n'a été détectée entre deux protéines) et matrice de Markov associée.

Cette méthode calcule la probabilité de marches aléatoires dans le graphe en alternant deux opérateurs nommés expansion et inflation. Le graphe est représenté sous la forme de sa matrice d'incidence transformée en une matrice de Markov (donc des probabilités des chemins dans le graphe), dont la somme sur chaque colonne est égale à un et dont les valeurs sont positives (voir figure 8.7). Les opé-

rateurs d'inflation et d'expansion permettent de recalculer cette dernière matrice et donc de simuler des marches aléatoires dans le graphe. L'opérateur d'inflation Γ est :

$$(\Gamma, M)_{pq} = \frac{(M_{pq})^r}{\sum_{i=1}^k (M_{iq})^r} \quad (8.4)$$

où M est la matrice de Markov, k le nombre d'éléments dans le graphe, p la ligne et q la colonne, et r un coefficient. Si r était égal à un, chaque valeur de la matrice serait donc simplement rapportée à la somme de sa colonne (elle deviendrait donc une probabilité si elle ne l'était pas déjà). Lorsque r est supérieur à 1, les chemins les plus probables sont favorisés au dépend des chemins les moins probables. Lors de l'étape d'expansion, des marches aléatoires sont effectuées en prenant le carré de la matrice plusieurs fois. Lors de ces marches, il est plus probable de rester dans le même groupe que de passer d'un groupe à l'autre. Ainsi, intervenant entre ces marches, l'opérateur d'inflation peut séparer les groupes car il a pour effet d'augmenter les probabilités les plus fortes (chemins intra-groupe) et de diminuer les plus faibles (chemins extra-groupes) et donc de séparer et renforcer les groupes, et d'autant plus que r est grand.

Le paramètre r influe très largement sur le nombre de groupes finaux déterminés (ou la granularité). Il est conseillé de le faire varier pour déterminer le paramètre convenant le mieux aux données. Ses valeurs sont en général entre 1,2 et 5. Plus ce paramètre est grand plus la granularité est fine, *i.e.* plus le nombre de groupe est grand et la taille des groupes est petite.

Les partitions obtenues peuvent en théorie être chevauchantes mais il faut en réalité de très fortes symétries dans le graphe pour que ce soit le cas.

Les classifications effectuées sont évaluées selon différents critères calculés par MCL comme l'efficacité - valeur entre 0 et 1 - et le nombre de groupes. L'efficacité est liée à deux autres valeurs, la fraction de masse (*mass fraction*) et la fraction de superficie (*area fraction*). La fraction de masse d'un groupe est la somme des poids des arêtes (définis dans le graphe initial) dont les deux sommets sont dans le groupe, rapporté au poids total des arêtes. La fraction de superficie est le carré de la taille des tous les groupes divisé par le carré du nombre total de sommets (en réalité $N(N-1)$ et non N^2 , N étant le nombre total de sommets).

Un autre critère est utile, la distance entre plusieurs partitions. Pour deux partitions effectuées à partir du même graphe mais pour des paramètres r différents, il est possible de trouver toutes les protéines qu'il faut changer de groupe pour transformer une partition en une sous-partition de l'autre. Soient par exemple 7 protéines numérotées de 1 à 7 et deux partitions dont les groupes sont $\{(1247), (356)\}$ pour la première et $\{(14), (2356), (7)\}$ pour la seconde. Pour que la première partition devienne une sous-partition de la seconde, il suffit que la protéine 2 soit déplacée dans le second groupe. Pour que la seconde partition devienne une sous-partition de la première, il faut déplacer les protéines 7 et 2 dans le premier groupe. Le nombre de protéines à déplacer est donc de 1 dans le premier cas et de 2 dans le deuxième, la différence entre ces deux partitions est de $(2+1)/7 * 2 = 0,21$.

Il faut donc déplacer en moyenne 20% des protéines pour transformer une partition en une autre.

8.2.2 Sensibilité aux paramètres et à la longueur des protéines

Les classifications en familles structurales ont été réalisées sur deux jeux de données : toutes les protéines de la banque non redondante et toutes les protéines de la banque de plus de 80 résidus.

Les premières classifications, pour un paramètre r compris entre 1,2 et 3 ont une efficacité comprise entre 0,41 (pour $r = 1,2$) et 0,7 (pour $r = 3$). Le nombre de groupes varie entre 725 (dont 230 groupes ne contenant qu'une protéine - ou singltons) et 3091 (dont 1115 singltons). Le problème de ces partitions est qu'elles contiennent toutes quelques familles très peuplées mais dont les protéines ne semblent pas structuralement similaires. Ces familles contiennent beaucoup de petites protéines notamment celles contenant beaucoup d'hélices α , ce qui semble attirer les grandes structures contenant beaucoup d'hélices α . Par exemple, pour $r = 1,8$, la première famille contient 498 protéines plutôt de petite taille (en moyenne 77 résidus) mais 10% des structures de cette famille ont plus de 160 résidus.

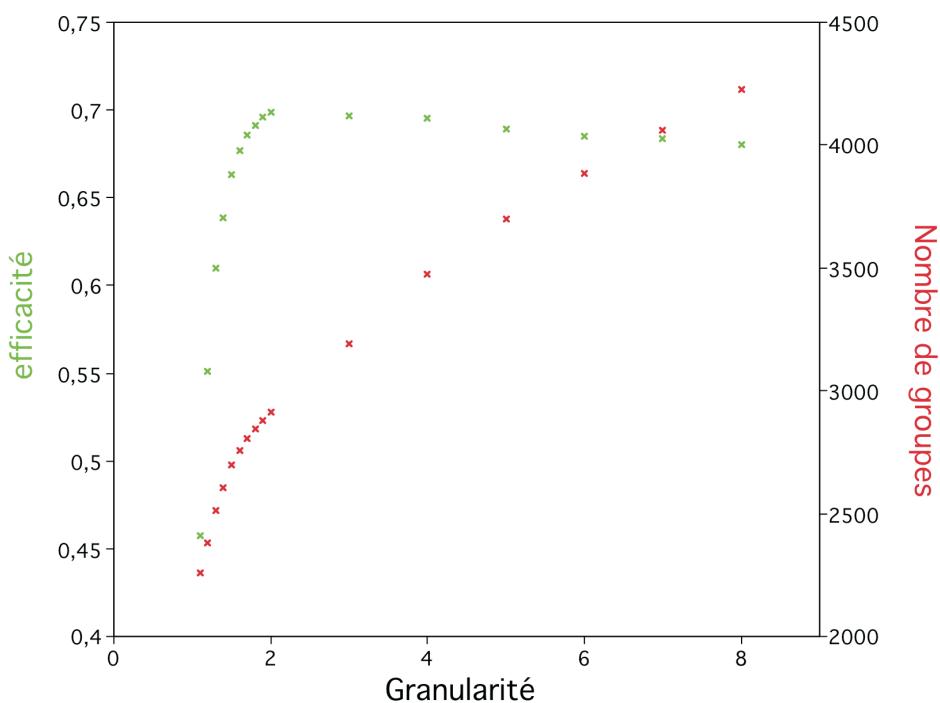


FIG. 8.8 – Efficacité et nombre de groupes pour les partitions du graphe ne contenant que les protéines de plus de 80 résidus. Le paramètre r varie entre 1,1 et 8. La meilleure efficacité est obtenue pour $r = 2$. Le nombre de groupes est alors de 2911 dont 1699 contiennent au moins 2 structures.

Les secondes classifications ont donc été effectuées uniquement pour les protéines de plus de 80 résidus. L'efficacité cette fois ne croît pas toujours avec le nombre de groupes. Les meilleurs résultats ont été obtenus avec un paramètre r de 2 (voir figure 8.8), et une efficacité meilleure que celles obtenues lors des premières classifications. La différence maximale entre deux partitions, atteinte entre

les partitions faites pour $r = 1,1$ et $r = 8$, est de 0,28 et il suffit de déplacer 12 protéines de la première partition pour obtenir une sous-partition de la seconde. Toutes les partitions sont assez semblables et assez peu de protéines sont classées dans des groupes contenant des protéines différentes lorsque r change.

Comme l'efficacité est meilleure pour $r = 2$, cette partition a été retenue. Sans compter les protéines seules, il y a 1700 familles (voir figure 8.9). La famille la plus peuplée contient 223 structures et seules 4 familles ont plus de 100 protéines. La taille moyenne des familles est de 6,6 structures, 50% des familles contenant 3 protéines ou moins. Comme un défaut de MCL est de découper en petites familles, il est probable que certaines de ces familles puissent être rassemblées. Ce nombre est cependant du même ordre de grandeur que le nombre de *superfamily* de SCOP (1539) et le nombre de familles de niveau H de CATH (1467).

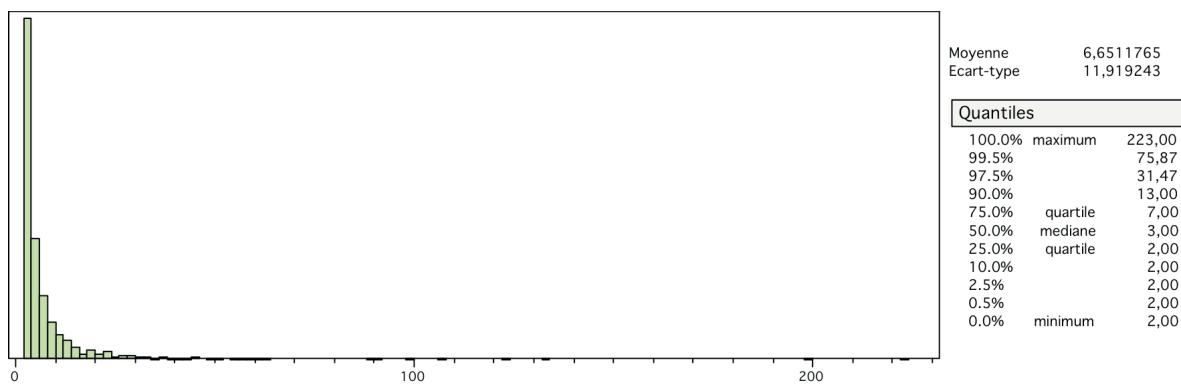


FIG. 8.9 – Distribution de la taille des familles déterminées pour les structures de plus de 80 résidus et pour $r = 2$.

8.2.3 Quelques exemples de familles structurales

Les deux classifications pour lesquelles quelques analyses ont été faites sont : la meilleure des classifications des protéines de plus de 80 résidus ($r = 2$) et la classification de toutes les protéines de la banque obtenue pour $r = 1,8$.

L'analyse complète des familles constituées n'a pas été faite, cependant certaines hypothèses sont étayées par quelques exemples.

La première hypothèse est que, les protéines regroupées dans une même famille étant structuralement similaires, elles partagent une fonction similaire. Ainsi, la première famille de la classification des protéines de plus de 80 résidus, contenant 223 protéines, est cohérente car toutes sont des globines. Un autre exemple est la famille des cytochromes P450 qui ne contient aucun autre type de protéine et qui contient tous les cytochromes P450 présents dans la banque. Une partie de graphe ne présentant que les protéines directement liées aux cytochromes P450 ainsi que les 35 cytochromes est présentée figure 8.10. Il se peut que certaine protéines soient « mal classées » mais il sera peut être possible de les identifier en recherchant les protéines qui changent de groupe entre deux partitions obtenues avec

des valeurs de r différentes. Il est aussi probable que certaines familles puissent être rassemblées en une seule.

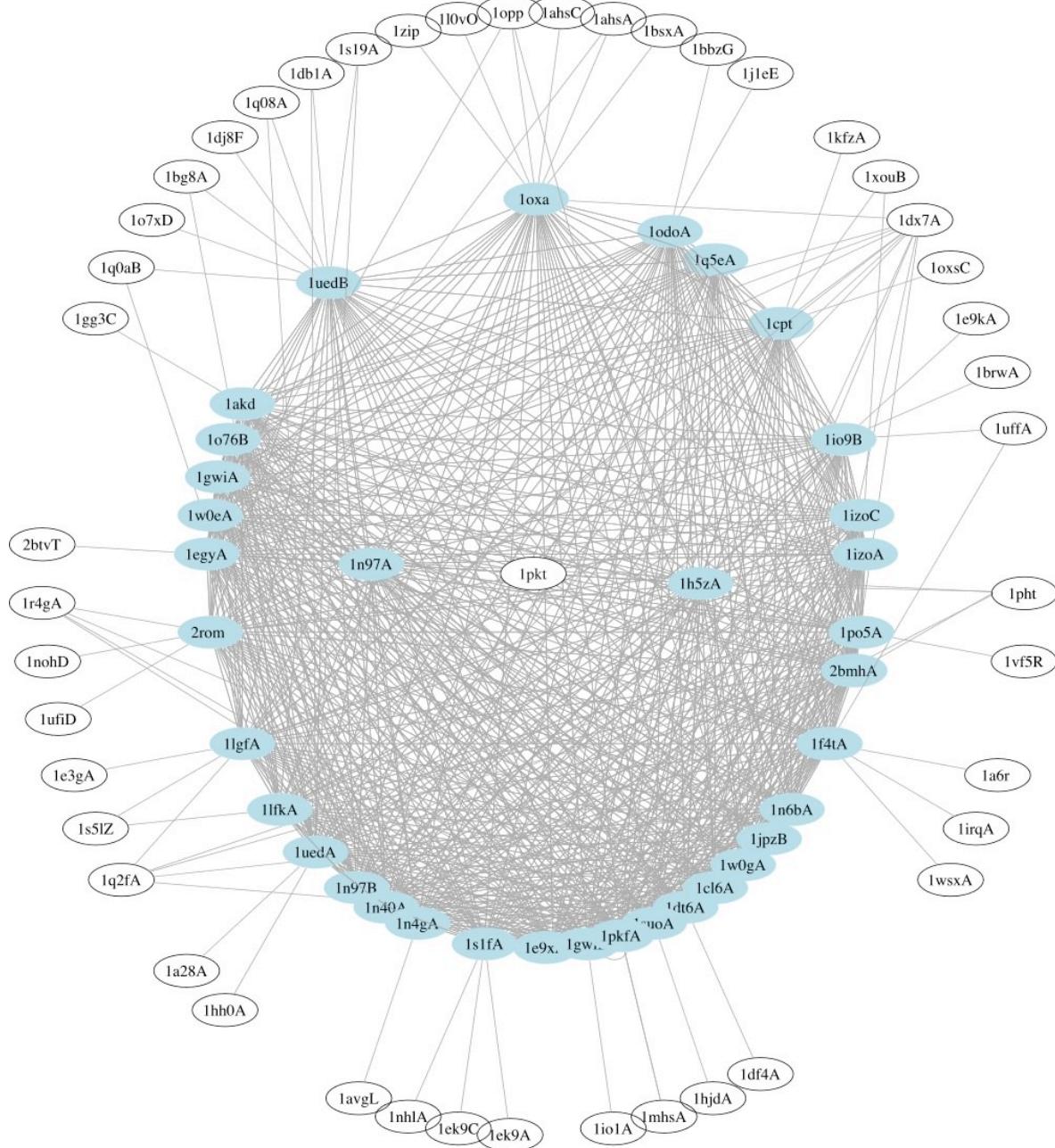


FIG. 8.10 – Représentation partielle du graphe des protéines de plus de 80 résidus. Seuls les cytochromes P450 (en bleu) et les protéines qui y sont directement liées dans le graphe sont représentées (en blanc). Seuls les 35 P450 sont présents dans la famille définie par MCL. La protéine blanche au centre est positionnée à cet endroit car elle n'est liée qu'aux deux cytochromes eux mêmes au centre. Il est visible que les cytochromes sont beaucoup plus connectés entre eux qu'avec les autres protéines.

Les structures ici classées n'ont pas été découpées en domaines. Si une protéine contient plusieurs domaines et que chacun est similaire à différentes protéines - appartenant à plusieurs classes, la pro-

téine contenant plusieurs domaines sera cependant classée dans une seule famille. Néanmoins, des liens sont indiqués dans les partitions trouvées par MCL lorsque une protéine est aussi liée à un autre groupe, ce qui permet de retrouver une telle particularité.

8.3 Constitution des « coeurs »

Les familles structurales étant constituées, l'objectif est alors de déterminer quels sont les blocs structuraux conservés parmi les structures d'une même famille. Deux des trois méthodes de comparaison multiple de structures ont été appliquées : les *m*-diagonales et le Gibbs sampling. Les alignements ont été effectués pour la meilleure des classifications, celle obtenue à partir des protéines de plus de 80 résidus ($r = 2$). Tous les alignements multiples présentés dans la partie « Nouvelles Méthodes » sont des alignements effectués sur la famille des cytochrome P450. Quelques analyses préliminaires ont été effectuées sur les blocs structuraux déterminés par la méthode dérivée du *Gibbs sampling* (la méthode des *m*-diagonales ne permettant pas d'aligner plus de 50 structures en un temps raisonnable, *i.e.* inférieur à 1 heure). Elle permet de comparer rapidement un très grand nombre de structures. Ainsi, même les 223 immunoglobulines de la première famille ont été comparées. Cependant, seuls 3 blocs structuraux, soit 24 résidus communs sont trouvés dans toutes les structures. Pour comparer un grand nombre de structures, il faudra donc assouplir la méthode, (*i.e.* utiliser des écarts types supérieurs à 3).

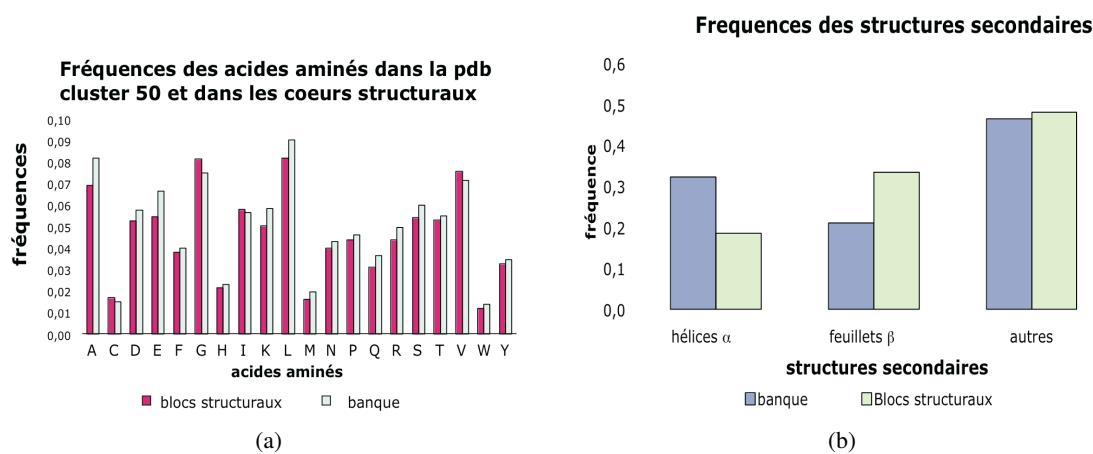


FIG. 8.11 – Compositions en acides aminés et structures secondaires des blocs structuraux déterminés par la méthode du *Gibbs sampling*.

La composition en acides aminés des blocs structuraux trouvés par la méthode du *Gibbs sampling* est présentée figure 8.11. Aucune variation notable n'est observée entre la composition en acides aminés de toutes les structures de la banque et celle des blocs structuraux. Par contre, la composition en structures secondaires varie, les blocs comportent moins d'hélices α et plus de feuilllets β que les structures.

La méthode du *Gibbs sampling* est très sensible aux protéines ne partageant pas ou peu de similarité structurale avec les autres structures de la même famille car elle aligne toujours toutes les structures. Par contre, la méthode des m-diagonales pouvant ne pas aligner certaines des structures (quorum), elle devrait être moins sensible à ces protéines mal classées. Des analyses ultérieures nous le diront.

Ces blocs structuraux conservés pourront constituer une base de cœurs utile pour les méthodes de reconnaissance de replis comme FROST (Marin et al., 2002). L'étape suivante sera d'analyser les cœurs obtenus notamment au niveau de leur composition en structures secondaires (« recouvrements » et corrélation) et de leur rigidité (facteurs de température).

8.4 Conclusion et perspectives

L'ensemble des résultats préliminaires indique que la procédure de classification semble correcte. Néanmoins, plusieurs points sont à améliorer.

Lors de l'étape d'identification des similarités entre paires de structures, la procédure de classification par les lois Normales multivariées peut être améliorée en calculant non plus la probabilité d'appartenir à la distribution mais la probabilité de trouver des valeurs supérieures à celles trouvées pour une paire de protéines données. En effet, pour une mesure comme le Z-score par exemple, l'éloignement par rapport à la moyenne des Z-scores de la classe « paires similaires » peut signifier ou bien une paire non similaire (Z-score << moyenne) ou bien une paire au contraire très similaire (Z-score >> moyenne). L'effet n'est pas dramatique dans la mesure où la paire très similaire a une probabilité quasi nulle d'appartenir à la classe des non-similaires.

De plus, il faudra tester les SVM de manière plus systématique.

Ensuite, lors de la détermination des familles, il faudra vérifier que les protéines multidomaines peuvent être attribuées à plusieurs classes. Si ce n'est pas le cas, il sera possible d'essayer d'autres méthodes permettant la définition de familles chevauchantes comme celle développée par C. Brun, C. Hermann et A. Guénoche (Brun et al., 2004).

Enfin, il faudra procéder à des analyses systématiques des familles trouvées en comparant les différentes partitions obtenues pour identifier les structures dont la famille change suivant la partition et rassembler certaines familles. Il serait aussi intéressant d'attribuer une fonction générale aux familles. Il est aussi bien sûr indispensable de comparer la classification ou les classifications finales aux autres classifications existantes (dans la mesure où elles sont comparables).

Quatrième partie

Conclusion et perspectives

L'objet de ce travail a été d'étudier les similarités structurales locales parmi les protéines dont la structure est connue, et plusieurs méthodes de comparaison de structures protéiques ont été développées.

La première méthode permet de comparer une structure avec toutes les structures d'une banque. D'après les tests effectués et par rapport aux résultats des autres méthodes de comparaison structurale, les similarités structurales locales déterminées par cette méthode permettent d'identifier les protéines similaires parmi les structures d'une banque. La description du squelette protéique par les angles α est donc suffisante et l'algorithme mis en place est efficace. De plus, l'indépendance vis-à-vis des structures secondaires permet d'identifier des similarités entre structures qui ne sont pas détectées par d'autres méthodes. Cette méthode a donné lieu à une publication présentant la méthode générale de recherche des similarités structurales locales et la comparaison des résultats obtenus avec ceux d'autres méthodes (Carpentier et al., 2005). Cependant, l'algorithme de la construction de l'automate n'a pas été encore publié (il se trouve en Annexe). Cette méthode a été mise à disposition sur la plateforme de Ressource Parisienne en Bioinformatique Structurale (RPBS) *via* une interface permettant une meilleure visualisation des résultats (Alland et al., 2005).

Les deux premières méthodes d'alignement multiple de structures sont tout à fait fonctionnelles et permettent la comparaison massive de structures. Elles ont d'ailleurs été appliquées à l'alignement des structures présentes dans les familles déterminées par classement des résultats de YAKUSA. La méthode des m-diagonales a pour avantage de pouvoir ne pas prendre en compte certaines structures : un bloc structural commun peut n'être présent que sur une partie des structures (*i.e.* selon un certain quorum). De plus, si une protéine ne partage aucune similarité structurale avec toutes les autres, elle peut être ignorée. Cette méthode a cependant pour désavantage de ne pas pouvoir aligner plus d'une cinquantaine de structures et de ne construire les blocs structuraux multiples qu'à partir des blocs structuraux déterminés entre les paires de structures. La seconde méthode d'alignement, dérivée du *Gibbs sampling*, a pour avantage de ne pas se baser sur les comparaisons de paires de structures. De plus, elle permet d'aligner un grand nombre de structures. Elle a cependant pour désavantage de ne détecter les blocs structuraux similaires que s'ils sont présents sur toutes les structures (quorum de 100%).

La troisième méthode de comparaison multiple repose sur un algorithme plus complexe. Cet algorithme est générique et pourra être appliqué à d'autres problèmes de recherche de motifs. Il a donné lieu à plusieurs publications (Pisanti et al., 2005a; Pisanti et al., 2005b; Pisanti et al., 2005c). Les avantages sont que tous les blocs structuraux respectant les paramètres sont exhaustivement trouvés et que la recherche de motifs est réellement multiple. Cette méthode est cependant trop lourde pour comparer systématiquement toutes les structures des familles déterminées.

Les résultats obtenus par ces méthodes multiples doivent encore être comparés à ceux des autres méthodes de comparaison multiple de structures et la méthodologie pour le faire est donc à mettre en place. En effet, on a vu que l'alignement optimal de deux (ou plus) structures n'est pas bien défini, et il n'est donc pas trivial de comparer les alignements obtenus par des méthodes différentes. Comparer

les alignements en termes de longueur et de $RMSD_c$ ne semble pas toujours adéquat.

Le développement de ces méthodes a soulevé plusieurs problèmes relatifs à l'identification des similarités structurales et à la comparaison des structures.

Le premier problème a été de définir le champ de la comparaison : doit-elle être globale ou locale ? La recherche d'une similarité globale est adaptée aux structures homologues proches, mais seule la recherche de similarités locales permet d'identifier des structures plus lointaines. La comparaison de toutes les structures connues ne peut donc s'appuyer que sur des méthodes de recherche de similarités locales. Toutes les méthodes développées ici sont des méthodes locales.

Un deuxième problème est de choisir quelles propriétés comparer entre les structures. Il a été choisi de prendre en compte uniquement les propriétés géométriques des structures tridimensionnelles des protéines. Ce choix semble raisonnable car les positions dans l'espace des atomes sont déterminées par leurs interactions. Comparer leurs positions permet donc de prendre en compte ces interactions. Le problème suivant a été de choisir le niveau de description auquel se placer. Faut-il comparer tous les atomes ? seuls ceux des squelettes carbonés ou C_α ? ou uniquement les structures secondaires ? Le niveau "tout atome" est trop précis pour comparer. Les méthodes travaillant aux deux autres niveaux fournissent des résultats complémentaires. En effet, les méthodes de comparaison de structures secondaires permettent d'identifier des similarités invisibles au niveau des C_α et, inversement, les méthodes de comparaison des squelettes carbonés permettent de d'identifier des similarités indécelables au niveau des structures secondaires car mettant en jeu des structures apériodiques. L'idée sous-jacente à mon travail étant la recherche de similarités locales, j'ai choisi la comparaison au niveau des C_α , aussi bien pour la comparaison deux à deux (YAKUSA, page 98) que pour les méthodes d'alignement multiple (pages 126 et 133).

La procédure de classification des structures protéiques en familles respecte toutes les conditions que nous nous étions imposées, sauf une. Elle est non hiérarchique et réalisée totalement automatiquement mais ne permet pour l'instant pas à une protéine composée de plusieurs domaines d'appartenir à plusieurs classes. Plusieurs solutions sont envisageables. La première est de chercher parmi les structures classées celles qui sont aussi liées à un ou plusieurs autres familles. La seconde est d'utiliser une autre méthode de classification des structures permettant le chevauchement des classes obtenues.

Les résultats préliminaires indiquent que la procédure de classification est adéquate. Le nombre de familles structurales définies est cohérent avec l'état des connaissances (*i.e.* le nombre de familles définies dans d'autres classifications). Les exemples observés de familles de structures semblent aussi confirmer cette adéquation car les protéines réunies dans une même classe partagent une même fonction. Cependant, un travail plus poussé indiquerait probablement que certaines protéines sont mal classées et que certaines familles pourraient être regroupées. Ces améliorations effectuées, il faudra alors comparer cette classification aux autres classifications existantes.

Les familles structurales étant définies, il sera alors possible de déterminer les blocs structuraux conservés. L'alignement multiple préliminaire de toutes les structures de chacune des familles a été effectué avec les deux premières méthodes de comparaison multiple de structures développées, la

méthode des *m*-diagonales (page 126) et celle du *Gibbs Sampling* appliquée aux structures 133).

Les blocs structuraux définis pourront d'abord constituer une base de cœurs pour la méthode de reconnaissance de repliements FROST (Marin et al., 2002). Cette nouvelle définition des cœurs devrait permettre d'améliorer les performances de la méthode et d'utiliser celle ci pour d'annotation structurale des génomes.

De plus, l'analyse des blocs structuraux conservés permettra de répondre à différentes questions. Les résidus d'un site actif sont-ils toujours localisés dans des blocs structuraux conservés ? Les blocs structuraux sont-ils situés dans des régions dont les facteurs de température sont faibles (régions « froides ») ? Peut-on trouver des résidus invariants ? La réponse à ces questions permettrait en retour qu'une analyse de séquences puisse fournir des réponses à des questions structurales sur des séquences dont la structure n'est pas connue.

Enfin, comparer les variations géométriques des blocs structuraux d'une même famille - éventuellement avec celles des séquences associées - pourrait donner des indices quant à l'établissement d'une mesure de distance évolutive au niveau des structures.

Beaucoup de personnes m'ont dit et répété qu'un travail de thèse n'est jamais achevé. J'ajouterais ici qu'il me semble à peine commencé...

Bibliographie, annexes et index

Bibliographie

- Aho, A. and Corasick, H. (1975). Efficient string matching : an aid to bibliographic search. *Comm. ACM*, 18(6) :333–340.
- Alberts, B., Bray, D., Lewis, J., Raff, M., Toberts, K., and Watson, J. (1994). *Molecular Biology of the Cell*. arland Publishing, Inc. New York, London.
- Alexandrov, N. (1996). Surfing the pdb. *Protein Eng.*, 9(9) :727–732.
- Alexandrov, N. N. and Fischer, D. (1996). Analysis of topological and nontopological structural similarities in the pdb : new examples with old structures. *Proteins*, 25(3) :354–65.
- Alexandrov, N. N. and Go, N. (1994). Biological meaning, statistical significance, and classification of local spatial similarities in nonhomologous proteins. *Protein Sci*, 3(6) :866–75.
- Alexandrov, N. N., Takahashi, K., and Go, N. (1992). Common spatial arrangements of backbone fragments in homologous and non-homologous proteins. *J Mol Biol*, 225(1) :5–9.
- Allard, C., Moreews, F., Boens, D., Carpentier, M., Chiusa, S., Lonquety, M., Renault, N., Wong, Y., Cantalloube, H., Chomilier, J., Hochez, J., Pothier, J., Villoutreix, B. O., Zagury, J. F., and Tuffery, P. (2005). Rpbs : a web resource for structural bioinformatics. *Nucleic Acids Res*, 33(Web Server issue) :W44–9.
- Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local alignment search tool. *J Mol Biol*, 215(3) :403–10.
- Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped blast and psi-blast : a new generation of protein database search programs. *Nucleic Acids Res*, 25(17) :3389–402.
- Anfinsen, C. B. (1973). Principles that govern the folding of protein chains. *Science*, 181(96) :223–30.
- Argos, P. (1987). A sensitive procedure to compare amino acid sequences. *J Mol Biol*, 193(2) :385–96.
- Artymiuk, P. J., Bath, P. A., Grindley, H. M., Pepperrell, C. A., Poirrette, A. R., Rice, D. W., Thorner, D. A., Wild, D. J., Willett, P., Allen, F. H., and et al. (1992). Similarity searching in databases of three-dimensional molecules and macromolecules. *J Chem Inf Comput Sci*, 32(6) :617–30.
- Artymiuk, P. J., Poirrette, A. R., Grindley, H. M., Rice, D. W., and Willett, P. (1994). A graph-theoretic approach to the identification of three-dimensional patterns of amino acid side-chains in protein structures. *J Mol Biol*, 243(2) :327–44.

- Artymiuk, P. J., Rice, D. W., Mitchell, E. M., and Willett, P. (1990). Structural resemblance between the families of bacterial signal-transduction proteins and of g proteins revealed by graph theoretical techniques. *Protein Eng*, 4(1) :39–43.
- Bachar, O., Fischer, D., Nussinov, R., and Wolfson, H. (1993). A computer vision based technique for 3-d sequence-independent structural comparison of proteins. *Protein Eng*, 6(3) :279–88.
- Balaji, S., Sujatha, S., Kumar, S. S., and Srinivasan, N. (2001). Pali-a database of phylogeny and alignment of homologous protein structures. *Nucleic Acids Res*, 29(1) :61–5.
- Balasubramanian, R. (1977). New type of representation for mapping chain-folding in protein molecules. *Nature*, 266(5605) :856–7.
- Barakat, M. T. and Dean, P. M. (1991). Molecular structure matching by simulated annealing. iii. the incorporation of null correspondences into the matching problem. *J Comput Aided Mol Des*, 5(2) :107–17.
- Barrow, H. G. and Burstal, R. M. (1976). Subgraph isomorphism, matching relational structures and maximal cliques. *Information Processing Letters*, 4(4) :83–84.
- Barton, G. J. and Sternberg, M. J. (1987). A strategy for the rapid multiple alignment of protein sequences. confidence levels from tertiary structure comparisons. *J Mol Biol*, 198(2) :327–37.
- Barton, G. J. and Sternberg, M. J. E. (1988). Lopal and scamp : techniques for the comparison and display of protein structures. *Journal of Molecular Graphics*, 6(4) :190–196.
- Baum, L., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *Ann. Math. Stat.*, 41 :164–71.
- Berchold, A. and Raftery, A. (1999). The mixture transition distribution (mtd) model for high-order markov chains and non-gaussian time series. Technical Report 360, University of Washington.
- Berge, J. M. F. T. (1977). Orthogonal procrustes rotation for two or more matrices. *Psychometrika*, 42 :267–76.
- Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. (2000). The protein data bank. *Nucleic Acids Res*, 28(1) :235–42.
- Bernstein, F. C., Koetzle, T. F., Williams, G. J., E. F. Meyer, J., Brice, M. D., Rodgers, J. R., Kennard, O., Shimanouchi, T., and Tasumi, M. (1977). The protein data bank. a computer-based archival file for macromolecular structures. *Eur J Biochem*, 80(2) :319–24.
- Blankenbecler, R., Ohlsson, M., Peterson, C., and Ringner, M. (2003). Matching protein structures with fuzzy alignments. *Proc Natl Acad Sci U S A*, 100(21) :11936–40.
- Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In Haussler, D., editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152. ACM Press.
- Boutonnet, N. S., Roonan, M. J., Ochagavia, M. E., Richelle, J., and Wodak, S. J. (1995). Optimal protein structure alignments by multiple linkage clustering : application to distantly related proteins. *Protein Eng*, 8(7) :647–62.
- Brandon, C. and Tooze, J. (1999). *Introduction to Protein Structure*. Garland Publishing Inc., New York, second edition edition.
- Brenner, S. E., Chothia, C., and Hubbard, T. J. (1998). Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proc Natl Acad Sci U S A*, 95(11) :6073–8.

- Brenner, S. E., Chothia, C., Hubbard, T. J., and Murzin, A. G. (1996). Understanding protein structure : using scop for fold interpretation. *Methods Enzymol*, 266 :635–43.
- Brenner, S. E., Koehl, P., and Levitt, M. (2000). The astral compendium for protein structure and sequence analysis. *Nucleic Acids Res*, 28(1) :254–6.
- Brint, A. and Willett, P. (1987). Algorithms for the identification of three-dimensional maximal common substructures. *J Chem Inf Comput Sci*, 27(4) :152–8.
- Bron, C. and Kerbosch, J. (1973). Finding all cliques of an undirected graph. *Comm. ACM*, 16 :575.
- Brown, N. P., Orengo, C. A., and Taylor, W. R. (1996). A protein structure comparison methodology. *Computers and Chemistry*, 20(3) :359–380.
- Brun, C., Herrmann, C., and Guenoche, A. (2004). Clustering proteins from interaction networks for the prediction of cellular functions. *BMC Bioinformatics*, 5(1) :95.
- Bucher, P. and Bairoch, A. (1994). A generalized profile syntax for biomolecular sequence motifs and its function in automatic sequence interpretation. *Proc Int Conf Intell Syst Mol Biol*, 2 :53–61.
- Camoglu, O., Kahveci, T., and Singh, A. K. (2003). Psi : indexing protein structures for fast similarity search. *Bioinformatics*, 19 Suppl 1 :i81–3.
- Camproux, A. C., Gautier, R., and Tuffery, P. (2004). A hidden markov model derived structural alphabet for proteins. *J Mol Biol*, 339(3) :591–605.
- Camproux, A. C., Tuffery, P., Chevrolat, J. P., Boisvieux, J. F., and Hazout, S. (1999). Hidden markov model approach for identifying the modular framework of the protein backbone. *Protein Eng*, 12(12) :1063–73.
- Carpentier, M., Brouillet, S., and Pothier, J. (2004). Alignement multiple et local de structures protéiques. In *JOBIM*, Montreal, Canada.
- Carpentier, M., Brouillet, S., and Pothier, J. (2005). Yakusa : a fast structural database scanning method. *Proteins*, 61(1) :137–51.
- Carugo, O. and Pongor, S. (2002). Protein fold similarity estimated by a probabilistic approach based on c(alpha)-c(alpha) distance comparison. *J Mol Biol*, 315(4) :887–98.
- Chew, L. P., Huttenlocher, D., Kedem, K., and Kleinberg, J. (1999). Fast detection of common geometric substructure in proteins. *J Comput Biol*, 6(3-4) :313–25.
- Chothia, C. (1984). Principles that determine the structure of proteins. *Annu Rev Biochem*, 53 :537–72.
- Chothia, C. (1992). Proteins. one thousand families for the molecular biologist. *Nature*, 357(6379) :543–4.
- Chothia, C. and Gerstein, M. (1997). Protein evolution. how far can sequences diverge ? *Nature*, 385(6617) :579, 581.
- Chothia, C. and Lesk, A. M. (1986). The relation between the divergence of sequence and structure in proteins. *The EMBO Journal*, 5(4) :823–826.
- Clark, D. E., Willett, P., and Kenny, P. W. (1991). Pharmacophoric pattern matching in files of three-dimensional chemical structures : use of smoothed bounded distances for incompletely specified query patterns. *J Mol Graph*, 9(3) :157–60.
- Coatney, M. and Parthasarathy, S. (2003). Motifminer : A general toolkit for efficiently identifying common substructures in molecules. In *ICDM '02 : Proceedings of the 2003 IEEE International Conference on Data Mining (ICDM'03)*. IEEE Computer Society.

- Cohen, F. E. and Sternberg, M. J. (1980). On the prediction of protein structure : The significance of the root-mean-square deviation. *J Mol Biol*, 138(2) :321–33.
- Cohen, G. (1997). Align : a program to superimpose protein coordinates, accounting for insertions and deletions. *Journal of Applied Crystallography*, 30(6) :1160–1161.
- Collobert, R. and Bengio, S. (2001). Svmtorch : Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1 :143–60.
- Cone, M. M., Venkataraghavan, R., and McLafferty, F. W. (1977). Molecular structure comparison program for the identification of maximal common substructures. *Journal of the American Chemical Society*, 99(23) :7668–71.
- Conte, L. L., Brenner, S. E., Hubbard, T. J., Chothia, C., and Murzin, A. G. (2002). Scop database in 2002 : refinements accommodate structural genomics. *Nucleic Acids Res*, 30(1) :264–7.
- Cook, D. J. and Holder, L. B. (1994). Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research*, 1 :231–55.
- Crandell, C. W. and Smith, D. H. (1983). Computer-assisted examination of compounds for common three-dimensional substructures. *J. Chem. Inf. Comput. Sci.*, 23(4) :186–97.
- Cristianini, N. and Shawe-Taylor, J. (2000). *An introduction to Support Vector Machines (and other kernel-based learning methods)*. Cambridge University Press, Cambridge.
- Cristobal, S., Zemla, A., Fischer, D., Rychlewski, L., and Elofsson, A. (2001). A study of quality measures for protein threading models. *BMC Bioinformatics*, 2(1) :5.
- Crochemore, M. and Rytter, W. (1994). *Text Algorithm*. Oxford University Press.
- Dayhoff, M. O., Barker, W. C., and Hunt, L. T. (1983). Establishing homologies in protein sequences. *Methods Enzymol*, 91 :524–45.
- Dayhoff, M. O., Eck, R. V., and Orcutt, B. C. (1978). A model of evolutionary change in proteins. In Dayhoff, M. O., editor, *Atlas of Protein Sequence and Structure*, pages 345–52. Nat. Biomed. Res. Found., Washington.
- de Brevern, A. G., Camproux, A. C., Hazout, S., Etchebest, C., and Tuffery, P. (2001). Protein structural alphabets : beyond the secondary structure description. In *Recent Adv. In Prot. Eng.*, pages 319–331. Research signpost, Trivandrum, India., sangadai sg ed. edition.
- de Rinaldis, M., Ausiello, G., Cesareni, G., and Helmer-Citterich, M. (1998). Three-dimensional profiles : a new tool to identify protein surface similarities. *J Mol Biol*, 284(4) :1211–21.
- Diamond, R. (1992). On the multiple simultaneous superposition of molecular structures by rigid body transformations. *Protein Sci*, 1(10) :1279–87.
- Diamond, R. (1995). Coordinate-based cluster analysis. *Acta Crystallographica Section D*, 51(2) :127–135.
- Dietmann, S., Park, J., Notredame, C., Heger, A., Lappe, M., and Holm, L. (2001). A fully automatic evolutionary classification of protein folds : Dali domain dictionary version 3. *Nucleic Acids Res*, 29(1) :55–7.
- Ding, D., Qian, J., and Feng, Z. (1994). A differential geometric treatment of protein structure comparison. *Bulletin of Mathematical Biology*, 56(5) :923–943.
- Dror, O., Benyamin, H., Nussinov, R., and Wolfson, H. (2003a). Mass : multiple structural alignment by secondary structures. *Bioinformatics*, 19(90001) :95i–104.

- Dror, O., Benyamin, H., Nussinov, R., and Wolfson, H. J. (2003b). Multiple structural alignment by secondary structures : algorithm and applications. *Protein Sci*, 12(11) :2492–507.
- Eidhammer, I., Jonassen, I., and Taylor, W. R. (2000). Structure comparison and structure patterns. *J Comput Biol*, 7(5) :685–716.
- Enright, A. J., Dongen, S. V., and Ouzounis, C. A. (2002). An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Res*, 30(7) :1575–84.
- Escalier, V. (1997). *Algorithmes pour la comparaison de structures moléculaires tridimensionnelles*. PhD thesis, Université Pierre et Marie Curie - Paris 6.
- Escalier, V., Pothier, J., Soldano, H., and Viari, A. (1998). Pairwise and multiple identification of three-dimensional common substructures in proteins. *J Comput Biol*, 5(1) :41–56.
- Esposito, L., Simone, A. D., Zagari, A., and Vitagliano, L. (2005). Correlation between omega and psi dihedral angles in protein structures. *J Mol Biol*, 347(3) :483–7.
- Estabrook, R. W. (2003). A passion for p450s (rememberances of the early history of research on cytochrome p450). *Drug Metab Dispos*, 31(12) :1461–73.
- Falicov, A. and Cohen, F. E. (1996). A surface of minimum area metric for the structural comparison of proteins. *J Mol Biol*, 258(5) :871–92.
- Felsenstein, J. (1985). Confidence limits on phylogenies : an approach using the bootstrap. *Evolution*, 39 :783–91.
- Feng, Z. K. and Sippl, M. J. (1996). Optimum superimposition of protein structures : ambiguities and implications. *Fold Des*, 1(2) :123–32.
- Fetrow, J. S. and Skolnick, J. (1998). Method for prediction of protein function from sequence using the sequence-to-structure-to-function paradigm with application to glutaredoxins/thioredoxins and t1 ribonucleases. *J Mol Biol*, 281(5) :949–68.
- Fischer, D., Bachar, O., Nussinov, R., and Wolfson, H. (1992). An efficient automated computer vision based technique for detection of three dimensional structural motifs in proteins. *J Biomol Struct Dyn*, 9(4) :769–89.
- Fischer, D., Barret, C., Bryson, K., Elofsson, A., Godzik, A., Jones, D., Karplus, K. J., Kelley, L. A., MacCallum, R. M., Pawowski, K., Rost, B., Rychlewski, L., and Sternberg, M. (1999). Cafasp-1 : critical assessment of fully automated structure prediction methods. *Proteins*, Suppl 3 :209–17.
- Fischer, D., Elofsson, A., Rice, D., and Eisenberg, D. (1996). Assessing the performance of fold recognition methods by means of a comprehensive benchmark. *Pac Symp Biocomput*, pages 300–18.
- Fischer, D. and Rychlewski, L. (2003). The 2002 olympic games of protein structure prediction. *Protein Eng*, 16(3) :157–60.
- Fischer, D., Wolfson, H., Lin, S. L., and Nussinov, R. (1994). Three-dimensional, sequence order-independent structural comparison of a serine protease against the crystallographic database reveals active site similarities : potential implications to evolution and to protein folding. *Protein Sci*, 3(5) :769–78.
- Flores, T. P., Moss, D. S., and Thornton, J. M. (1994). An algorithm for automatically generating protein topology cartoons. *Protein Eng*, 7(1) :31–7.
- Flory, P. J. (1969). *Statistical Mechanics of Chain Molecules*. Wiley, New York.

- Garey, M. R. and Johnson, M. S. (1979). *Computers and intractability : A guide to the theory of NPcompleteness*. W.H. Freeman and company, New York.
- Gerber, P. R. and Muller, K. (1987). Superimposing several sets of atomic coordinates. *Acta Crystallographica Section A*, 43(3) :426–428.
- Gerstein, M. and Altman, R. (1995). Using a measure of structural variation to define a core for the globins. *Comput. Appl. Biosci.*, 11(6) :633–644.
- Gerstein, M. and Levitt, M. (1996). Using iterative dynamic programming to obtain accurate pairwise and multiple alignments of protein structures. In *Proc Int Conf Intell Syst Mol Biol*, volume 4, pages 59–67.
- Gerstein, M. and Levitt, M. (1998). Comprehensive assessment of automatic structural alignment against a manual standard, the scop classification of proteins. *Protein Sci*, 7(2) :445–56.
- Gibrat, J. F., Madej, T., and Bryant, S. H. (1996). Surprising similarities in structure comparison. *Curr Opin Struct Biol*, 6(3) :377–85.
- Gilbert, D., Westhead, D., Nagano, N., and Thornton, J. (1999). Motif-based searching in tops protein topology databases. *Bioinformatics*, 15(4) :317–26.
- Gilbert, D., Westhead, D., Viksna, J., and Thornton, J. (2001). A computer system to perform structure comparison using tops representations of protein structure. *Comput Chem*, 26(1) :23–30.
- Glover, F. (1989). Tabu search - part i. *ORSA Journal on Computing*, 1 :190–260.
- Godzik, A. (1996). The structural alignment between two proteins : is there a unique answer ? *Protein Sci*, 5(7) :1325–38.
- Govindarajan, S., Recabarren, R., and Goldstein, R. A. (1999). Estimating the total number of protein folds. *Proteins*, 35(4) :408–14.
- Gowri, V. S., Pandit, S. B., Karthik, P. S., Srinivasan, N., and Balaji, S. (2003). Integration of related sequences with protein three-dimensional structural families in an updated version of pali database. *Nucleic Acids Res*, 31(1) :486–8.
- Gribskov, M., McLachlan, A. D., and Eisenberg, D. (1987). Profile analysis : detection of distantly related proteins. *Proc Natl Acad Sci U S A*, 84(13) :4355–8.
- Gries, D. (1982). A Note on a Standard Strategy for Developing Loop Invariants and Loops. *Science of Computer Programming*, 2 :207–14.
- Grindley, H. M., Artymiuk, P. J., Rice, D. W., and Willett, P. (1993). Identification of tertiary structure resemblance in proteins using a maximal common subgraph isomorphism algorithm. *J Mol Biol*, 229(3) :707–21.
- Guda, C., Lu, S., Scheeff, E. D., Bourne, P. E., and Shindyalov, I. N. (2004). Ce-mc : a multiple protein structure alignment server. *Nucleic Acids Res*, 32(Web Server issue) :W100–3.
- Guda, C., Scheeff, E. D., Bourne, P. E., and Shindyalov, I. N. (2001). A new algorithm for the alignment of multiple protein structures using monte carlo optimization. In *Pacific Symposium on Biocomputing*, volume 6, pages 275–286, Hawaii.
- Gumbel, E. (1958). *Statistics of Extremes*. Columbia University Press.
- Guyon, F., Camproux, A. C., Hochez, J., and Tuffery, P. (2004). Sa-search : a web tool for protein structure mining based on a structural alphabet. *Nucleic Acids Res*, 32(Web Server issue) :W545–8.

- Haraldsson, H. and Ohlsson, M. (2004). A fuzzy matching approach to multiple structure alignment of proteins.
- Harrison, A., Pearl, F., Sillitoe, I., Slidel, T., Mott, R., Thornton, J., and Orengo, C. (2003). Recognizing the fold of a protein structure. *Bioinformatics*, 19(14) :1748–1759.
- Ho, C. M. and Marshall, G. R. (1993). Foundation : a program to retrieve all possible structures containing a user-defined minimum number of matching query elements from three-dimensional databases. *J Comput Aided Mol Des*, 7(1) :3–22.
- Hobohm, U., Scharf, M., Schneider, R., and Sander, C. (1992). Selection of representative protein data sets. *Protein Sci*, 1(3) :409–17.
- Holliday, J. D. and Willett, P. (1997). Using a genetic algorithm to identify common structural features in sets of ligands. *J Mol Graph Model*, 15(4) :221–32.
- Holm, L. and Sander, C. (1993). Protein structure comparison by alignment of distance matrices. *J Mol Biol*, 233(1) :123–38.
- Holm, L. and Sander, C. (1994a). The fssp database of structurally aligned protein fold families. *Nucl. Acids Res.*, 22(17) :3600–3609.
- Holm, L. and Sander, C. (1994b). Parser for protein folding units. *Proteins*, 19(3) :256–68.
- Holm, L. and Sander, C. (1994c). Searching protein structure databases has come of age. *Proteins*, 19(3) :165–73.
- Holm, L. and Sander, C. (1995a). 3-D lookup : fast protein structure database searches at 90reliability. *Proc Int Conf Intell Syst Mol Biol*, 3 :179–87.
- Holm, L. and Sander, C. (1995b). Dali : a network tool for protein structure comparison. *Trends Biochem Sci*, 20(11) :478–80.
- Holm, L. and Sander, C. (1996a). Alignment of three-dimensional protein structures : network server for database searching. *Methods Enzymol*, 266 :653–62.
- Holm, L. and Sander, C. (1996b). The fssp database : fold classification based on structure-structure alignment of proteins. *Nucleic Acids Res*, 24(1) :206–9.
- Holm, L. and Sander, C. (1996c). Mapping the protein universe. *Science*, 273(5275) :595–603.
- Hutchinson, E. G. and Thornton, J. M. (1996). Promotif—a program to identify and analyze structural motifs in proteins. *Protein Sci*, 5(2) :212–20.
- Jakes, S. E. and Willett, P. (1986). Pharmacophoric pattern matching in files of 3-d chemical structures : election of interatomic distance screens. *J. Mol. Graph.*, 4(1) :12–20.
- Jambon, M., Imbert, A., Deleage, G., and Geourjon, C. (2003). A new bioinformatic approach to detect common 3d sites in protein structures. *Proteins*, 52(2) :137–45.
- Jean, P., Pothier, J., Dansette, P. M., Mansuy, D., and Viari, A. (1997). Automated multiple analysis of protein structures : application to homology modeling of cytochromes p450. *Proteins*, 28(3) :388–404.
- Johnson, M. S., Sali, A., and Blundell, T. L. (1990a). Phylogenetic relationships from three-dimensional protein structures. *Methods Enzymol*, 183 :670–90.
- Johnson, M. S., Sutcliffe, M. J., and Blundell, T. L. (1990b). Molecular anatomy : phyletic relationships derived from three-dimensional structures of proteins. *J Mol Evol*, 30(1) :43–59.

- Jonassen, I., Eidhammer, I., Grindhaug, S. H., and Taylor, W. R. (2000). Searching the protein structure databank with weak sequence patterns and structural constraints. *J Mol Biol*, 304(4) :599–619.
- Jonassen, I., Eidhammer, I., and Taylor, W. R. (1999). Discovery of local packing motifs in protein structures. *Proteins*, 34(2) :206–19.
- Jung, J. and Lee, B. (2000). Protein structure alignment using environmental profiles. *Protein Eng*, 13(8) :535–43.
- Kabsch, W. (1976). A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica*, A32 :922–923.
- Kabsch, W. (1978). A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 34(5) :827–828.
- Kabsch, W. and Sander, C. (1983). Dictionary of protein secondary structure : pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12) :2577–637.
- Karlin, S. and Altschul, S. F. (1990). Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc Natl Acad Sci U S A*, 87(6) :2264–8.
- Karp, R., R.E., M., and Rosenberg, A. (1972). Rapid identification of repeated patterns in strings, trees and arrays. In *Fourth ACM Symposium on Theory of Computing*, pages 125–36.
- Karpen, M. E., de Haseth, P. L., and Neet, K. E. (1989). Comparing short protein substructures by a method based on backbone torsion angles. *Proteins*, 6(2) :155–67.
- Kasuya, A. and Thornton, J. M. (1999). Three-dimensional structure analysis of prosite patterns. *J Mol Biol*, 286(5) :1673–91.
- Kawabata, T. (2003). Matras : A program for protein 3d structure comparison. *Nucleic Acids Res*, 31(13) :3367–9.
- Kawabata, T. and Nishikawa, K. (2000). Protein structure comparison using the markov transition model of evolution. *Proteins*, 41(1) :108–22.
- Kearsley, S. (1989). On the orthogonal transformation used for structural comparisons. *Acta Crystallographica Section A*, 45(2) :208–210.
- Kearsley, S. K. (1990). An algorithm for the simultaneous superposition of a structural series. *Journal of Computational Chemistry*, 11(10) :1187 – 1192.
- Kedem, K., Chew, L. P., and Elber, R. (1999). Unit-vector rms (urms) as a tool to analyze molecular dynamics trajectories. *Proteins*, 37(4) :554–64.
- Kelley, L. A., MacCallum, R., and Sternberg, M. (1999). Recognition of remote protein homologies using three-dimensional information to generate a position specific scoring matrix in the program 3d-pssm. In Istrail, S., Pevzner, P., and Waterman, M., editors, *RECOMB*, pages 218–225, Lyon. The Association for Computing Machinery, New York.
- Kelley, L. A., MacCallum, R. M., and Sternberg, M. J. (2000). Enhanced genome annotation using structural profiles in the program 3d-pssm. *J Mol Biol*, 299(2) :499–520.
- Kendrew, J. C., Bodo, G., Dintzis, H. M., Parrish, R. G., Wyckoff, H., and Phillips, D. C. (1958). A three-dimensional model of the myoglobin molecule obtained by x-ray analysis. *Nature*, 181(4610) :662–6.
- Kihara, D. and Skolnick, J. (2003). The pdb is a covering set of small protein structures. *J Mol Biol*, 334(4) :793–802.

- Kleywegt, G. (1996). Use of non-crystallographic symmetry in protein structure refinement. *Acta Crystallographica Section D*, 52(4) :842–857.
- Kleywegt, G., Lamerichs, R., Boelens, R., and Kaptein, R. (1989). Toward automatic assignment of protein h nmr spectra. *Journal of Magnetic Resonance*, 85(1) :186–197.
- Kleywegt, G. J. (1999). Recognition of spatial motifs in protein structures. *J Mol Biol*, 285(4) :1887–97.
- Kleywegt, G. J. and Jones, T. A. (1995). Where freedom is given, liberties are taken. *Structure*, 3(6) :535–40.
- Kleywegt, G. J. and Jones, T. A. (1997). Detecting folding motifs and similarities in protein structures. In Sweet, C. W. C. J. and M., R., editors, *Methods in Enzymology ; Macromolecular Crystallography Part B*, volume 277, pages 525–545. Academic Press.
- Knuth, D., Jr, J. M., and Pratt, V. (1977). Fast pattern matching in strings. *SIAM Journal on Computing*, 6(1) :323–350.
- Koch, I., Kaden, F., and Selbig, J. (1992). Analysis of protein sheet topologies by graph theoretical methods. *Proteins*, 12(4) :314–23.
- Koch, I., Lengauer, T., and Wanke, E. (1996). An algorithm for finding maximal common subtopologies in a set of protein structures. *J Comput Biol*, 3(2) :289–306.
- Koehl, P. (2001). Protein structure similarities. *Curr Opin Struct Biol*, 11(3) :348–53.
- Kolodny, R., Koehl, P., and Levitt, M. (2005). Comprehensive evaluation of protein structure alignment methods : Scoring by geometric measures. *Journal of Molecular Biology*, 346(4) :1173–1188.
- Kraulis, P. (1991). Molscript : a program to produce both detailed and schematic plots of protein structures. *Journal of Applied Crystallography*, 24(5) :946–950.
- Krissinel, E. and Henrick, K. (2003). Protein structure comparison in 3d based on secondary structure matching (ssm) followed by ca alignment, scored by a new structural similarity function. In Kungl, A. J. K. and J., P., editors, *Proceedings of the 5th International Conference on Molecular Structural Biology*, page 88, Vienna.
- Krissinel, E. and Henrick, K. (2004a). Common subgraph isomorphism detection by backtracking search. *Software : Practice and Experience*, 34(6) :591–607.
- Krissinel, E. and Henrick, K. (2004b). Secondary-structure matching (ssm), a new tool for fast protein structure alignment in three dimensions. *Acta Crystallographica Section D*, 60(12 Part 1) :2256–2268.
- Kyte, J. and Doolittle, R. F. (1982). A simple method for displaying the hydropathic character of a protein. *J Mol Biol*, 157(1) :105–32.
- Lackner, P., Koppensteiner, W. A., Sippl, M. J., and Domingues, F. S. (2000). Prosup : a refined tool for protein structure alignment. *Protein Eng*, 13(11) :745–52.
- Lamdan, Y., Schwarttz, J., and Wolfson, H. (1988). On recognition of 3-d objects from 2-d images. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 1407–1413 vol.3.
- Lamdan, Y. and Wolfson, H. (1988). Geometric hashing : A general and efficient model-based recognition scheme. In *Computer Vision., 1988. Second International Conference on*, pages 238–249.

- Lathrop, R. H. (1994). The protein threading problem with sequence amino acid interaction preferences is np-complete. *Protein Eng*, 7(9) :1059–68.
- Lawrence, C. E., Altschul, S. F., Boguski, M. S., Liu, J. S., Neuwald, A. F., and Wootton, J. C. (1993). Detecting subtle sequence signals : a gibbs sampling strategy for multiple alignment. *Science*, 262(5131) :208–14.
- Leibowitz, N., Fligelman, Z. Y., Nussinov, R., and Wolfson, H. J. (1999). Multiple structural alignment and core detection by geometric hashing. *Proc Int Conf Intell Syst Mol Biol*, pages 169–77.
- Leibowitz, N., Fligelman, Z. Y., Nussinov, R., and Wolfson, H. J. (2001). Automated multiple structure alignment and detection of a common substructural motif. *Proteins*, 43(3) :235–45.
- Leluk, J., Konieczny, L., and Roterman, I. (2003). Search for structural similarity in proteins. *Bioinformatics*, 19(1) :117–24.
- Lesk, A. (1986). A toolkit for computational molecular biology. ii. on the optimal superposition of two sets of coordinates. *Acta Crystallographica Section A*, 42(2) :110–113.
- Lesk, A. M. (1979). Detection of three-dimensional patterns of atoms in chemical structures. *Commun. ACM*, 22(4) :219–24.
- Lesk, A. M. (1991). *Protein Architecture*. Oxford University Press, New York.
- Levi, G. (1972). A note on the derivation of maximal common subgraphs of two directed or undirected graphs. *Calcolo*, 9 :341–52.
- Levine, M., Stuart, D., and Williams, J. (1984). A method for the systematic comparison of the three-dimensional structures of proteins and some results. *Acta Crystallographica Section A*, 40 :600–610.
- Levitt, M. (1976). A simplified representation of protein conformations for rapid simulation of protein folding. *J Mol Biol*, 104(1) :59–107.
- Levitt, M. and Chothia, C. (1976). Structural patterns in globular proteins. *Nature*, 261(5561) :552–8.
- Levitt, M. and Gerstein, M. (1998). A unified statistical framework for sequence comparison and structure comparison. *Proc Natl Acad Sci U S A*, 95(11) :5913–20.
- Levitt, M. and Warshel, A. (1975). Computer simulation of protein folding. *Nature*, 253(5494) :694–8.
- Lieberman, M. N. (1982). *Molecular Structure and Biological Activity*. Molecular Structure and Biological Activity. Elsevier, New York.
- Lu, G. (2000). Top : a new method for protein structure comparisons and similarity searches. *Journal of Applied Crystallography*, 33(1) :176–183.
- Lupyan, D., Leo-Macias, A., and Ortiz, A. R. (2005). A new progressive-iterative algorithm for multiple structure alignment. *Bioinformatics*.
- MacArthur, M. W. and Thornton, J. M. (1996). Deviations from planarity of the peptide bond in peptides and proteins. *J Mol Biol*, 264(5) :1180–95.
- Madej, T., Gibrat, J. F., and Bryant, S. H. (1995). Threading a database of protein cores. *Proteins*, 23(3) :356–69.
- Madsen, D. and Kleywegt, G. J. (2002). Interactive motif and fold recognition in protein structures. *Journal of Applied Crystallography*, 35(1) :137–139.
- Marin, A., Malliavin, T. E., Nicolas, P., and Delsuc, M. A. (2004). From nmr chemical shifts to amino acid types : investigation of the predictive power carried by nuclei. *J Biomol NMR*, 30(1) :47–60.

- Marin, A., Pothier, J., Zimmermann, K., and Gibrat, J. F. (2002). Frost : a filter-based fold recognition method. *Proteins*, 49(4) :493–509.
- Martin, A. C. (2000). The ups and downs of protein topology ; rapid comparison of protein structure. *Protein Eng*, 13(12) :829–37.
- Matsuda, H., Taniguchi, F., and Hashimoto, A. (1997). An approach to detection of protein structural motifs using an encoding scheme of backbone conformations. *Pac Symp Biocomput*, pages 280–91.
- Matsuo, Y. and Kanehisa, M. (1993). An approach to systematic detection of protein structural motifs. *Comput Appl Biosci*, 9(2) :153–9.
- May, A. C. (1999). Toward more meaningful hierarchical classification of protein three-dimensional structures. *Proteins*, 37(1) :20–9.
- May, A. C. and Johnson, M. S. (1994). Protein structure comparisons using a combination of a genetic algorithm, dynamic programming and least-squares minimization. *Protein Eng*, 7(4) :475–85.
- May, A. C. and Johnson, M. S. (1995). Improved genetic algorithm-based protein structure comparisons : pairwise and multiple superpositions. *Protein Eng*, 8(9) :873–82.
- McLachlan, A. (1982). Rapid comparison of protein structures. *Acta Crystallographica Section A*, 38(6) :871–873.
- McLachlan, A. D. (1979). Gene duplication in the evolution of the yeast hexokinase active site. *Eur J Biochem*, 100(1) :181–7.
- Michie, A. D., Orengo, C. A., and Thornton, J. M. (1996). Analysis of domain structural class using an automated class assignment protocol. *J Mol Biol*, 262(2) :168–85.
- Mitchell, E. M., Artymiuk, P. J., Rice, D. W., and Willett, P. (1990). Use of techniques derived from graph theory to compare secondary structure motifs in proteins. *J Mol Biol*, 212(1) :151–66.
- Mizuguchi, K., Deane, C. M., Blundell, T. L., and Overington, J. P. (1998). Homstrad : a database of protein structure alignments for homologous families. *Protein Sci*, 7(11) :2469–71.
- Mizuguchi, K. and Go, N. (1995). Comparison of spatial arrangements of secondary structural elements in proteins. *Protein Eng*, 8(4) :353–62.
- Murzin, A. G., Brenner, S. E., Hubbard, T., and Chothia, C. (1995). Scop : a structural classification of proteins database for the investigation of sequences and structures. *J Mol Biol*, 247(4) :536–40.
- Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*, 48(3) :443–53.
- Neuwald, A. F., Liu, J. S., and Lawrence, C. E. (1995). Gibbs motif sampling : detection of bacterial outer membrane protein repeats. *Protein Sci*, 4(8) :1618–32.
- Nishikawa, K., Momany, F., and Scheraga, H. (1974). Low-energy structures of two dipeptides and their relationship to bend conformations. *Macromolecules*, 7(6) :797–806.
- Nishikawa, K. and Ooi, T. (1974). Comparison of homologous tertiary structures of proteins. *J Theor Biol*, 43(2) :351–74.
- Novotny, M., Madsen, D., and Kleywegt, G. J. (2004). Evaluation of protein fold comparison servers. *Proteins*, 54(2) :260–70.
- Nussinov, R. and Wolfson, H. (1991). Efficient detection of three-dimensional structural motifs in biological macromolecules by computer vision techniques. *PNAS*, 88(23) :10495–10499.

- Ochagavia, M., Richelle, J., and Wodak, S. (2002). Advanced pairwise structure alignments of proteins and analysis of conformational changes. *Bioinformatics*, 18(4) :637–640.
- Ochagavia, M. and Wodak, S. (2004). Progressive combinatorial algorithm for multiple structural alignments : Application to distantly related proteins. *Proteins : Structure, Function, and Bioinformatics*, 55(2) :436–454.
- Ogata, K., Ohya, M., and Umeyama, H. (1998). Amino acid similarity matrix for homology modeling derived from structural alignment and optimized by the monte carlo method. *J Mol Graph Model*, 16(4-6) :178–89, 254.
- Oldfield, T. J. and Hubbard, R. E. (1994). Analysis of c alpha geometry in protein structures. *Proteins*, 18(4) :324–37.
- Orengo, C. A. (1999). Cora-topological fingerprints for protein structural families. *Protein Sci*, 8(4) :699–715.
- Orengo, C. A., Brown, N. P., and Taylor, W. R. (1992). Fast structure alignment for protein databank searching. *Proteins*, 14(2) :139–67.
- Orengo, C. A., Jones, D. T., and Thornton, J. M. (1994). Protein superfamilies and domain superfolds. *Nature*, 372(6507) :631–4.
- Orengo, C. A., Michie, A. D., Jones, S., Jones, D. T., Swindells, M. B., and Thornton, J. M. (1997). Cath—a hierachic classification of protein domain structures. *Structure*, 5(8) :1093–108.
- Orengo, C. A. and Taylor, W. R. (1990). A rapid method of protein structure alignment. *J Theor Biol*, 147(4) :517–51.
- Orengo, C. A. and Taylor, W. R. (1993). A local alignment method for protein structure motifs. *J Mol Biol*, 233(3) :488–97.
- Orengo, C. A. and Taylor, W. R. (1996). Ssap : sequential structure alignment program for protein structure comparison. *Methods Enzymol*, 266 :617–35.
- Ortiz, A. R., Strauss, C. E., and Olmea, O. (2002). Mammoth (matching molecular models obtained from theory) : an automated method for model comparison. *Protein Sci*, 11(11) :2606–21.
- Overington, J. P., Zhu, Z. Y., Sali, A., Johnson, M. S., Sowdhamini, R., Louie, G. V., and Blundell, T. L. (1993). Molecular recognition in protein families : a database of aligned three-dimensional structures of related proteins. *Biochem Soc Trans*, 21 (Pt 3)(3) :597–604.
- Padlan, E. and Davies, D. (1975). Variability of three-dimensional structure in immunoglobulins. *Proc Natl Acad Sci USA*, 72(3) :819–823.
- Park, J., Karplus, K., Barrett, C., Hughey, R., Haussler, D., Hubbard, T., and Chothia, C. (1998). Sequence comparisons using multiple sequences detect three times as many remote homologues as pairwise methods. *J Mol Biol*, 284(4) :1201–10.
- Parthasarathy, S. and Coatney, M. (2002). Efficient discovery of common substructures in macromolecules. In *ICDM '02 : Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)*, page 362. IEEE Computer Society.
- Pearl, F., Todd, A., Sillitoe, I., Dibley, M., Redfern, O., Lewis, T., Bennett, C., Marsden, R., Grant, A., Lee, D., Akpor, A., Maibaum, M., Harrison, A., Dallman, T., Reeves, G., Diboun, I., Addou, S., Lise, S., Johnston, C., Sillero, A., Thornton, J., and Orengo, C. (2005). The cath domain structure database and related resources gene3d and dhs provide comprehensive domain family information for genome analysis. *Nucleic Acids Res*, 33(Database issue) :D247–51.

- Pearl, F. M., Lee, D., Bray, J. E., Sillitoe, I., Todd, A. E., Harrison, A. P., Thornton, J. M., and Orengo, C. A. (2000). Assigning genomic sequences to cath. *Nucleic Acids Res*, 28(1) :277–82.
- Pearl, F. M., Martin, N., Bray, J. E., Buchan, D. W., Harrison, A. P., Lee, D., Reeves, G. A., Shepherd, A. J., Sillitoe, I., Todd, A. E., Thornton, J. M., and Orengo, C. A. (2001). A rapid classification protocol for the cath domain database to support structural genomics. *Nucleic Acids Res*, 29(1) :223–7.
- Pearson, W. R. (1991). Searching protein sequence libraries : comparison of the sensitivity and selectivity of the smith-waterman and fasta algorithms. *Genomics*, 11(3) :635–50.
- Pearson, W. R. (2000). Flexible sequence similarity searching with the fasta3 program package. *Methods Mol Biol*, 132 :185–219.
- Pearson, W. R. and Lipman, D. J. (1988). Improved tools for biological sequence comparison. *Proc Natl Acad Sci U S A*, 85(8) :2444–8.
- Pennec, X. and Ayache, N. (1994). n $O(n^2)$ algorithm for 3D substructure matching of proteins. *Bioinformatics*, 14(6) :516–522.
- Pennec, X. and Ayache, N. (1998). A geometric algorithm to find small but highly similar 3d substructures in proteins. *Bioinformatics*, 14(6) :516–22.
- Pepperrell, C. A. and Willett, P. (1991). Techniques for the calculation of three-dimensional structural similarity using inter-atomic distances. *J Comput Aided Mol Des*, 5(5) :455–74.
- Petitjean, M. (1998). Interactive maximal common 3d substructure searching with the combined sdm/rms algorithm. *Computers and Chemistry*, 22(6) :463–465.
- Phillips, D. C. (1970). Development of crystallographic enzymology. In *Biochem. Soc. Symp.*, volume 31, page 11–28.
- Pisanti, N., Soldano, H., and Carpentier, M. (2005a). Incremental inference of relational motifs with a degenerate alphabet. In Alberto Apostolico, Maxime Crochemore, K. P., editor, *Combinatorial Pattern Matching : 16th Annual Symposium, CPM 2005, Jeju Island, Korea, 2005. Proceedings*, volume 3537 of *Lecture Notes in Computer Science*, pages 229–240, Jeju Island, Korea. Springer-Verlag GmbH.
- Pisanti, N., Soldano, H., Carpentier, M., and Pothier, J. (2005b). Implicit and explicit representation of approximated motifs. In Iliopoulos, C., Park, K., and K., S., editors, *Algorithms for Bioinformatics, C*, volume in press. King's College London Press, London.
- Pisanti, N., Soldano, H., Carpentier, M., and Pothier, J. (2005c). Inference of approximated motifs with conserved relations. *Journal of Discrete Algorithms*, submitted.
- Plewczynski, D., Pas, J., Grotthuss, M. V., and Rychlewski, L. (2004). Comparison of proteins based on segments structural similarity. *Acta Biochim Pol*, 51(1) :161–72.
- Plewczynski, D., Pas, J., von Grotthuss, M., and Rychlewski, L. (2002). 3d-hit : fast structural comparison of proteins. *Appl Bioinformatics*, 1(4) :223–5.
- Poirrette, A. R., Artymiuk, P. J., Rice, D. W., and Willett, P. (1997). Comparison of protein surfaces using a genetic algorithm. *J Comput Aided Mol Des*, 11(6) :557–69.
- Poirrette, A. R., Willett, P., and Allen, F. H. (1991). Pharmacophoric pattern matching in files of three-dimensional chemical structures : characterization and use of generalized valence angle screens. *J Mol Graph*, 9(4) :203–17.
- Rackovsky, S. and Goldstein, D. A. (1988). Protein comparison and classification : a differential geometric approach. *Proc Natl Acad Sci U S A*, 85(3) :777–81.

- Rackovsky, S. and Scheraga, H. (1978). Differential geometry and polymer conformation. 1. comparison of protein conformations. *Macromolecules*, 11(6) :1168–74.
- Rackovsky, S. and Scheraga, H. (1980). Differential geometry and polymer conformation. 2. development of a conformational distance function. *Macromolecules*, 13 :1440–53.
- Rackovsky, S. and Scheraga, H. (1981). Differential geometry and polymer conformation. 3. single-site and nearest-neighbor distributions, and nucleation of protein folding. *Macromolecules*, 14 :1259–69.
- Rackovsky, S. and Scheraga, H. (1982). Differential geometry and polymer conformation. 4. conformational and nucleation properties of individual amino acids. *Macromolecules*, 15 :1340–6.
- Raftery, A. (1985). A model for high-order markov chains. *Journal of the Royal Statistical Society, B*, 47 (3) :528–539.
- Ramachandran, G. N., Ramakrishnan, C., and Sesisakharan, V. (1963). Stereochemistry of polypeptide chain configurations. *J Mol Biol*, 7 :95–99.
- Rao, S. T. and Rossmann, M. G. (1973). Comparison of super-secondary structures in proteins. *J Mol Biol*, 76(2) :241–56.
- Raymond, J. W. and Willett, P. (2002). Maximum common subgraph isomorphism algorithms for the matching of chemical structures. *J Comput Aided Mol Des*, 16(7) :521–33.
- Remington, S. J. and Matthews, B. W. (1978). A general method to assess similarity of protein structures, with applications to t4 bacteriophage lysozyme. *Proc Natl Acad Sci U S A*, 75(5) :2180–4.
- Remington, S. J. and Matthews, B. W. (1980). A systematic approach to the comparison of protein structures. *J Mol Biol*, 140(1) :77–99.
- Richards, F. M. (1991). The protein folding problem. *Sci Am*, 264(1) :54–7, 60–3.
- Richardson, J. S. (1981). The anatomy and taxonomy of protein structure. *Adv Protein Chem*, 34 :167–339.
- Robson, B. (1999). Beyond proteins. *Trends Biotechnol*, 17(8) :311–5.
- Rogen, P. and Fain, B. (2003). Automatic classification of protein structure by using gauss integrals. *Proc Natl Acad Sci USA*, 100(1) :119–24.
- Rohlf, F. J. and Slice, D. E. (1990). Extensions of the procrustes method for the optimal superimposition of landmarks. *Syst. Zool.*, 39 :40–59.
- Rose, G. D. (1979). Hierarchic organization of domains in globular proteins. *Journal of Molecular Biology*, 134(3) :447–470.
- Rossmann, M. G. and Argos, P. (1975). A comparison of the heme binding pocket in globins and cytochrome b5. *J Biol Chem*, 250(18) :7525–32.
- Rossmann, M. G. and Argos, P. (1976). Exploring structural homology of proteins. *Journal of Molecular Biology*, 105(1) :75–95.
- Roterman, I. (1995). The geometrical analysis of peptide backbone structure and its local deformations. *Biochimie*, 77(3) :204–16.
- Rufino, S. D. and Blundell, T. L. (1994). Structure-based identification and clustering of protein families and superfamilies. *Journal of Computer-Aided Molecular Design (Historical Archive)*, 8(1) :5–27.
- Russell, R. B. and Barton, G. J. (1992). Multiple protein sequence alignment from tertiary structure comparison : assignment of global and residue confidence levels. *Proteins*, 14(2) :309–23.

- Rychlewski, L., Fischer, D., and Elofsson, A. (2003). Livebench-6 : large-scale automated evaluation of protein structure prediction servers. *Proteins*, 53 Suppl 6 :542–7.
- Sagot, M. F. (1996). *Ressemblance lexicale et structurale entre macromolecules - Formalisation et approches combinatoires*. PhD thesis, Universite de Marne-la-Vallée.
- Sagot, M. F., Viari, A., Pothier, J., and Soldano, H. (1995). Finding flexible patterns in a text : an application to three- dimensional molecular matching. *Comput Appl Biosci*, 11(1) :59–70.
- Sali, A. and Blundell, T. L. (1990). Definition of general topological equivalence in protein structures. a procedure involving comparison of properties and relationships through simulated annealing and dynamic programming. *J Mol Biol*, 212(2) :403–28.
- Sali, A. and Overington, J. P. (1994). Derivation of rules for comparative protein modeling from a database of protein structure alignments. *Protein Sci*, 3(9) :1582–96.
- Schmidt, R., Gerstein, M., and Altman, R. B. (1997). Lpfc : an internet library of protein family core structures. *Protein Sci*, 6(1) :246–8.
- Schuler, G. D., Altschul, S. F., and Lipman, D. J. (1991). A workbench for multiple alignment construction and analysis. *Proteins*, 9(3) :180–90.
- Schulz, G. E. (1977). Recognition of phylogenetic relationships from polypeptide chain fold similarities. *J Mol Evol*, 9 :339–42.
- Schulz, G. E. (1980). Gene duplication in glutathione reductase. *J Mol Biol*, 138(2) :335–47.
- Sedgewick, R. (1991). *Algorithmes en langage C*. Sciences sup. Dunod, dunod edition.
- Shapiro, A. and Botha, J. D. (1988). Dual algorithms for orthogonal procrustes rotations. *SIAM Journal on Matrix Analysis and Applications*, 9 :378–83.
- Shapiro, A., Botha, J. D., Pastore, A., and Lesk, A. M. (1992). A method for multiple superposition of structures. *Acta Crystallogr A*, 48 (Pt 1) :11–4.
- Shapiro, J. and Brutlag, D. (2004a). Foldminer and lock 2 : protein structure comparison and motif discovery on the web. *Nucleic Acids Res*, 32(Web Server issue) :W536–41.
- Shapiro, J. and Brutlag, D. (2004b). Foldminer : structural motif discovery using an improved superposition algorithm. *Protein Sci*, 13(1) :278–94.
- Shatsky, M., Nussinov, R., and Wolfson, H. (2002). Flexible protein alignment and hinge detection. *Proteins : Structure, Function, and Genetics*, 48(2) :242–256.
- Shatsky, M., Nussinov, R., and Wolfson, H. (2004). Flexprot : Alignment of flexible protein structures without a predefinition of hinge regions. *Journal of Computational Biology*, 11(1) :83–106.
- Sheridan, R. P., A. Rusinko, r., Nilakantan, R., and Venkataraghavan, R. (1989). Searching for pharmacophores in large coordinate data bases and its use in drug design. *Proc Natl Acad Sci U S A*, 86(20) :8165–9.
- Shindyalov, I. N. and Bourne, P. E. (1998). Protein structure alignment by incremental combinatorial extension (ce) of the optimal path. *Protein Eng*, 11(9) :739–47.
- Siddiqui, A. S. and Barton, G. J. (1995). Continuous and discontinuous domains : an algorithm for the automatic generation of reliable protein domain definitions. *Protein Sci*, 4(5) :872–84.
- Sierk, M. L. and Pearson, W. R. (2004). Sensitivity and selectivity in protein structure comparison. *Protein Sci*, 13(3) :773–85.
- Siew, N., Elofsson, A., Rychlewski, L., and Fischer, D. (2000). Maxsub : an automated measure for the assessment of protein structure prediction quality. *Bioinformatics*, 16(9) :776–85.

- Simons, K. T., Bonneau, R., Ruczinski, I., and Baker, D. (1999). Ab initio protein structure prediction of casp iii targets using rosetta. *Proteins, Suppl 3* :171–6.
- Singh, A. P. and Brutlag, D. L. (1997). Hierarchical protein structure superposition using both secondary structure and atomic representations. *Proc Int Conf Intell Syst Mol Biol*, 5 :284–93.
- Sippl, M. J. (1982). On the problem of comparing protein structures. development and applications of a new method for the assessment of structural similarities of polypeptide conformations. *J Mol Biol*, 156(2) :359–88.
- Sippl, M. J. (1990). Calculation of conformational ensembles from potentials of mean force. an approach to the knowledge-based prediction of local structures in globular proteins. *J Mol Biol*, 213(4) :859–83.
- Sippl, M. J. and Stegbuchner, H. (1991). Superposition of three-dimensional objects : A fast and numerically stable algorithm for the calculation of the matrix of optimal rotation. *Computers Chemistry*, 15(1) :73–78.
- Smith, T. F., Conte, L. L., Bienkowska, J., Gaitatzes, C., R. G. Rogers, J., and Lathrop, R. (1997). Current limitations to protein threading approaches. *J Comput Biol*, 4(3) :217–25.
- Smith, T. F. and Waterman, M. S. (1981). Identification of common molecular subsequences. *J Mol Biol*, 147(1) :195–7.
- Sneath, P. and Sokal, R. (1973). *Numerical Taxonomy- the principles and practice of numerical classification*. Freeman, San Francisco.
- Soldano, H., Viari, A., and Champesme, M. (1995). Searching for flexible repeated patterns using a non-transitive similarity relation. *Pattern Recognition Letters*, 16 :243–46.
- Sternberg, M. J. and Thornton, J. M. (1977). On the conformation of proteins : the handedness of the connection between parallel beta-strands. *J Mol Biol*, 110(2) :269–83.
- Stryer, L. (1994). *Biochemistry*. W.H. Freeman and Company, New York.
- Su, S., Cook, D. J., and Holder, L. B. (1999). Applications of knowledge discovery to molecular biology : identifying structural regularities in proteins. *Pac Symp Biocomput*, pages 190–201.
- Subbarao, N. and Haneef, I. (1991). Defining topological equivalences in macromolecules. *Protein Eng*, 4(8) :877–84.
- Subbiah, S., Laurens, D., and Levitt, M. (1993). Structural similarity of dna-binding domains of bacteriophage repressors and the globin core. *Current Biology*, 3(3) :141–148.
- Sujatha, S., Balaji, S., and Srinivasan, N. (2001). Pali : a database of alignments and phylogeny of homologous protein structures. *Bioinformatics*, 17(4) :375–6.
- Sutcliffe, M. J., Haneef, I., Carney, D., and Blundell, T. L. (1987). Knowledge based modelling of homologous proteins, part i : Three-dimensional frameworks derived from the simultaneous superposition of multiple structures. *Protein Eng*, 1(5) :377–84.
- Swindells, M. B. (1995). A procedure for detecting structural domains in proteins. *Protein Sci*, 4(1) :103–12.
- Swindells, M. G. and Alexandrov, N. N. (1994). Nucleotide binding in beta alpha beta–beta alpha beta topologies. *Nat Struct Biol*, 1(10) :677–8.
- Szustakowski, J. D. and Weng, Z. (2000). Protein structure alignment using a genetic algorithm. *Proteins*, 38(4) :428–40.

- Szustakowski, J. D. and Weng, Z. (2002). Protein structure alignment using evolutionary computing. In Kaufman, M., Fogel, G., and Corne, D., editors, *Evolutionary Computation in Bioinformatics*. Elsevier.
- Takayashi, Y., Maeda, S., and ich Sasaki, S. (1987). Automated recognition of common geometrical patterns among a variety of three-dimensional molecular structures. *Analytica Chimica Acta*, 200 :363–377.
- Taylor, W. (1999). Protein structure comparison using iterated double dynamic programming. *Protein Sci*, 8(3) :654–665.
- Taylor, W. R. (1988). A flexible method to align large numbers of biological sequences. *J Mol Evol*, 28(1-2) :161–9.
- Taylor, W. R. (1997). Random structural models for double dynamic programming score evaluation. *J Mol Evol*, 44(7) :174–80.
- Taylor, W. R., Flores, T. P., and Orengo, C. A. (1994). Multiple protein structure alignment. *Protein Sci*, 3(10) :1858–70.
- Taylor, W. R., May, A. C., Brown, N. P., and Aszodi, A. (2001). Protein structure : geometry, topology and classification. *Reports on Progress in Physics*, 64(4) :517.
- Taylor, W. R. and Orengo, C. A. (1989a). A holistic approach to protein structure alignment. *Protein Eng*, 2(7) :505–19.
- Taylor, W. R. and Orengo, C. A. (1989b). Protein structure alignment. *J Mol Biol*, 208(1) :1–22.
- Thompson, W., Rouchka, E. C., and Lawrence, C. E. (2003). Gibbs recursive sampler : finding transcription factor binding sites. *Nucleic Acids Res*, 31(13) :3580–5.
- Thornton, J., Jones, D., MacArthur, M., Orengo, C., and Swindells, M. B. (1995). Protein folds : towards understanding folding from inspection of native structures. *Philos. Trans. Roy. Soc. Lond. B Biol. Sci.*, 348 :71–9.
- Ullmann, J. R. (1976). An algorithm for subgraph isomorphism. *Journal of the ACM*, 23(1) :31–42.
- Unger, R., Harel, D., Wherland, S., and Sussman, J. L. (1989). A 3d building blocks approach to analyzing and predicting structure of proteins. *Proteins*, 5(4) :355–73.
- Usha, R. and Murthy, M. R. (1986). Protein structural homology : a metric approach. *Int J Pept Protein Res*, 28(4) :364–9.
- van Dongen, S. (2000). *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht.
- Vriend, G. (1990). What if : a molecular modeling and drug design program. *J Mol Graph*, 8(1) :52–6, 29.
- Vriend, G. and Sander, C. (1991). Detection of common three-dimensional substructures in proteins. *Proteins*, 11(1) :52–8.
- Wallace, A. C., Borkakoti, N., and Thornton, J. M. (1997). Tess : a geometric hashing algorithm for deriving 3d coordinate templates for searching structural databases. application to enzyme active sites. *Protein Sci*, 6(11) :2308–23.
- Wallace, A. C., Laskowski, R. A., and Thornton, J. M. (1996). Derivation of 3d coordinate templates for searching structural databases : application to ser-his-asp catalytic triads in the serine proteinases and lipases. *Protein Sci*, 5(6) :1001–13.
- Wang, G. and R. L. Dunbrack, J. (2003). Pisces : a protein sequence culling server. *Bioinformatics*, 19(12) :1589–91.

- Wang, Y. (2005). Pairwise protein structure comparison techniques.
- Wang, Z. X. (1996). How many fold types of protein are there in nature ? *Proteins*, 26(2) :186–91.
- Wang, Z. X. (1998). A re-estimation for the total numbers of protein folds and superfamilies. *Protein Eng*, 11(8) :621–6.
- Westhead, D. R., Slidel, T. W., Flores, T. P., and Thornton, J. M. (1999). Protein structural topology : Automated analysis and diagrammatic representation. *Protein Sci*, 8(4) :897–904.
- Wetlaufer, D. B. (1973). Nucleation, rapid folding, and globular intrachain regions in proteins. *Proc Natl Acad Sci U S A*, 70(3) :697–701.
- Wriggers, W. and Schulten, K. (1997). Protein domain movements : detection of rigid domains and visualization of hinges in comparisons of atomic coordinates. *Proteins : Structure, Function, and Genetics*, 29(1) :1–14.
- Wu, T. D., Schmidler, S. C., Hastie, T., and Brutlag, D. L. (1998a). Modeling and superposition of multiple protein structures using affine transformations : analysis of the globins. *Pac Symp Biocomput*, pages 509–20.
- Wu, T. D., Schmidler, S. C., Hastie, T., and Brutlag, D. L. (1998b). Regression analysis of multiple protein structures. *J Comput Biol*, 5(3) :585–95.
- Yang, A. S. and Honig, B. (1999). Sequence to structure alignment in comparative modeling using prism. *Proteins*, Suppl 3 :66–72.
- Yang, A. S. and Honig, B. (2000a). An integrated approach to the analysis and modeling of protein sequences and structures. i. protein structural alignment and a quantitative measure for protein structural distance. *J Mol Biol*, 301(3) :665–78.
- Yang, A. S. and Honig, B. (2000b). An integrated approach to the analysis and modeling of protein sequences and structures. iii. a comparative study of sequence conservation in protein structural families using multiple structural alignments. *J Mol Biol*, 301(3) :691–711.
- Ye, J. and Janardan, R. (2004). Approximate multiple protein structure alignment using the sum-of-pairs distance. *J Comput Biol*, 11(5) :986–1000.
- Ye, Y. and Godzik, A. (2003). Flexible structure alignment by chaining aligned fragment pairs allowing twists. *Bioinformatics*, 19 Suppl 2 :II246–II255.
- Ye, Y. and Godzik, A. (2004a). Database searching by flexible protein structure alignment. *Protein Sci*, 13(7) :1841–50.
- Ye, Y. and Godzik, A. (2004b). Fatcat : a web server for flexible structure comparison and structure similarity searching. *Nucleic Acids Res*, 32(Web Server issue) :W582–5.
- Ye, Y. and Godzik, A. (2005). Multiple flexible structure alignment using partial order graphs. *Bioinformatics*, 21(10) :2362–9.
- Yee, D. and Dill, K. (1993). Families and the structural relatedness among globular proteins. *Protein Sci*, 2 :884–9.
- Zemla, A. (2003). Lga : A method for finding 3d similarities in protein structures. *Nucleic Acids Res*, 31(13) :3370–4.
- Zhang, Y. and Skolnick, J. (2004). Scoring function for automated assessment of protein structure template quality. *Proteins*, 57(4) :702–10.
- Zhang, Y. and Skolnick, J. (2005). Tm-align : a protein structure alignment algorithm based on the tm-score. *Nucleic Acids Res*, 33(7) :2302–9.

- Zhu, J. and Weng, Z. (2005). Fast : A novel protein structure alignment algorithm. *Proteins*, 58(3) :618–27.
- Zhu, Z. Y., Sali, A., and Blundell, T. L. (1992). A variable gap penalty function and feature weights for protein 3-d structure comparisons. *Protein Eng*, 5(1) :43–51.
- Zuker, M. and Somorjai, R. L. (1989). The alignment of protein structures in three dimensions. *Bull Math Biol*, 51(1) :55–78.

ANNEXE A

Algorithmes connus utilisés en bioinformatique

A.1 La programmation dynamique NWS (Needleman and Wunsch, 1970).

Technique 1.

La programmation dynamique NWS (Needleman and Wunsch, 1970).

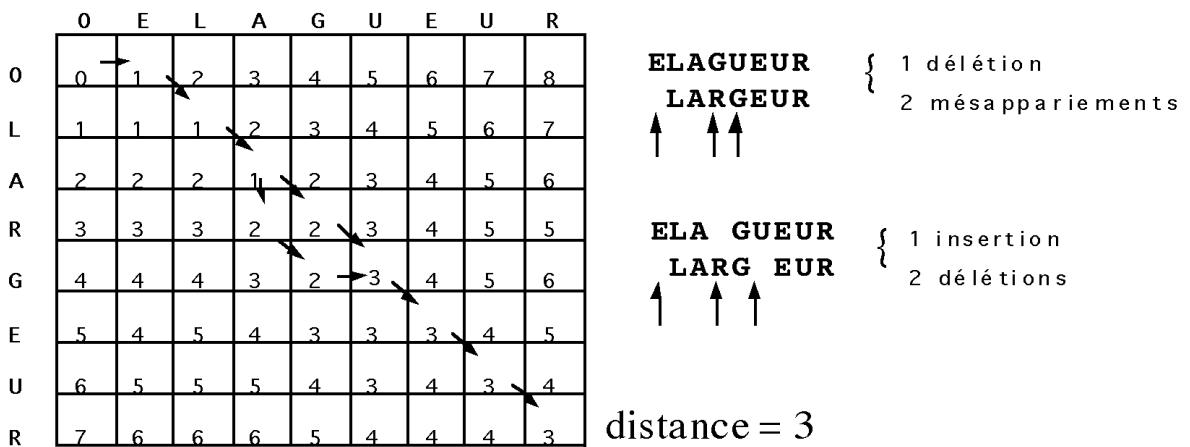


FIG. A.1 – Pour cet alignement, les coûts d'insertion, de déletion ou de substitution sont de 1. Note : quand on a le choix des chemins pour remonter entre indel ou substitution, on choisit toujours la substitution (cellule en diagonale), donc ici, l'alignement du haut.

Cet algorithme se décrit récursivement de la façon suivante : Soient deux séquences A et B de longueurs respectives m et n , on note a_i et b_j les résidus correspondant à $A = (a_1, \dots, a_n)$ et $B = (b_1, \dots, b_m)$. On note $D_{i,j}$ la distance minimum entre les deux séquences alignées du début jusqu'aux résidus a_i et b_j . On calcule successivement les $D_{i,j}$ pour des valeurs croissantes de i et j , jusqu'à atteindre la valeur de $D_{m,n}$ qui sera la distance minimum entre A et B . Les valeurs $D_{i,j}$ sont stockées dans un tableau à deux dimensions (matrice d'alignement). La procédure débute en $D_{0,0} = 0$. La valeur d'une cellule (i, j) est définie à partir des trois cellules précédentes $(i - 1, j)$, $(i - 1, j - 1)$ et $(i, j - 1)$. Ainsi on

calcule $D_{i,j}$ à partir de l'équation récursive suivante :

$$D_{i,j} = \min \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \\ \min_{1 \leq k \leq j} \{D_{i,j-k} + g(k)\} \\ \min_{1 \leq l \leq i} \{D_{i-l,j} + g(l)\} \end{cases} \quad (\text{A.1})$$

avec $w(a_i, b_j)$ le coût de la substitution de a_i par b_j et $g(k)$ une fonction « élastique » de pénalité de discontinuité (**gap**) $g(k) = a + (k - 1) \times b$ avec a la pénalité d'ouverture de gap et b l'incrément pour l'extension de gap. Seul le meilleur score est donc conservé dans la matrice, ainsi que le chemin correspondant, c'est-à-dire la cellule qui a fourni la meilleure distance. L'alignement des deux séquences est le meilleur chemin conduisant au meilleur score c.-a.-d celui de la dernière case (voir figure A.1).

A.2 *p-value, e-value et distribution des valeurs extrêmes*

La *p-value* est la probabilité d'obtenir par hasard le résultat d'une expérience. La *e-value* est le nombre attendu de fois qu'on pourrait obtenir le résultat d'une expérience par hasard.

Les scores maximaux d'alignement (dans certaines conditions) suivent une distribution des valeurs extrêmes (Gumbel, 1958; Karlin and Altschul, 1990). Elle a pour densité de probabilité :

$$\rho(s) = \kappa m n e^{-\lambda s} e^{-\kappa m n e^{-\lambda s}} \quad (\text{A.2})$$

avec s le score, m et n les longueurs des séquences, μ et κ des paramètres d'échelle (μ joue sur les scores, et κ sur les longueurs de séquences, donc l'espace de recherche). Connaissant ces deux valeurs, en intégrant la distribution ci dessus, il est possible de calculer la probabilité qu'un score soit supérieur à un seuil, probabilité qui est une *p-value* :

$$P(X > s) = 1 - e^{-\kappa m n e^{-\lambda s}} \quad (\text{A.3})$$

A.3 Algorithmes de Triades

Algorithm 1 Fonction principale : lecture des arguments, des données, initialisation et lancement de l'algorithme.

// Lecture des arguments...

Init()

DoKMREco() ou DoKMREcoDec() ou DoKMREcoChev()

// Lecture des données et création du premier vecteur V

// KMR dans ses différents incrément

Remarques pour l'algorithme 7 :

1. Pour les fonctions Push et Reinit() cf. algorithmes 4 et 5 respectivement ;
2. IsInclude(p1, p2) est une fonction qui renvoie Vrai si la pile p1 a tous ses éléments inclus dans la pile p2. Le test tient compte du fait que les éléments de p1 et p2 sont ordonnés ;
3. FusionABR() et InsersionABR sont des fonctions permettant respectivement la fusion des deux ABR et l'insersion d'une valeur dans un ABR (arbre binaire de recherche).

Algorithm 2 DoKMREco : Calcul de l'incrément et appel de la fonction de recherche/construction des mots.

Arguments d'entrée :

lgMot : la longueur initiale des motifs varie suivant les données (1 pour les séquences, 3 pour les structures).

lgMotVoulue : la longueur de motifs voulue (longueur maximale par défaut).

incr : de combien les motifs augmentent à chaque tour, c'est ce qui varie dans les différents DoKMREcoXXX(), il peut aussi être calculé dans le cas du chevauchement fixe (DoKMREcoChev())

Arguments d'entrée et de sortie :

v : vecteur de piles indexé sur la séquence ; à chaque position il y a les numéros de motifs commençant en cette position

Variables locales :

tABRPref : tableau indexé sur les numéros de motifs qui contient pour chaque motif un ABR contenant les préfixes de ce motif.

tABRSuff : de même mais pour les suffixes.

```
while v non vide et lgMot + incr < lgVoulue do           // Tant qu'il y a des motifs et qu'ils sont de taille inférieure à celle voulue
    ApplyLemmaOne(lgMot + incr, V, tABRSuf, tABRPref)      // Coeur de Triades
    lgMot ← lgMot +                                     // Cette instruction est modifiée dans les autres fonctions DoKMREcoXXX()
end while
```

Algorithm 3 ApplyLemmaOne : Appel des 4 fonctions représentant les 4 étapes de Triades (*cf. page 142*)

Arguments d'entrée :

lgNewMots : longueur des mots en train d'être construits.

Arguments d'entrée et de sortie :

v : *cf. algorithme 2*

tABRPref : *cf. algorithme 2*

tABRSuff : *cf. algorithme 2*

```
Qa ← BuildQa(v, , tABRPref, tABRSuf, lgNewMots)          //Constructions des mots de taille lgMots + incr.
Qb, tPS ← BuildQb(Qa, lgNewMots)                          // Ventilation de Qa dans Qb selon les relations ; Le quorum est en partie vérifié
v, tCorresGp ← BuildVab(Qb, lgNewMots)                    // Dépiler le vecteur Qa dans un nouveau vecteur v ; Le quorum est vérifié
v, tABRPref, tABRSuf ← FilterVab(v, tPS, tCorresGp)       // éliminer les motifs non maximaux.
Return(v, tABRPref, tABRSuf)
```

Algorithm 4 BuildQa : Construction des mots de taille lgMot+incr à partir des mots de taille lgMot ; ventilation et combinaison des éléments des piles deV dans la pile Qa

Arguments d'entrée :

lgNewMots : longueur des mots en train d'être construits.
 v : cf. algorithme 2
tABRPref : cf. algorithme 2
tABRSuff : cf. algorithme 2

Arguments de sortie :

Qa : vecteur de pile contenant à la fois la position et le numéro du préfixe ; il y a autant de pile dansQa que de motif dansV (les piles sont indexées sur les numéros de suffixes).

Variables locales :

pos : position dans la séquence
gpSuf et gpPref : numéro de motif des suffixes et préfixes

```

p=Dual(v)
for each pile qui a pour index gpPref dans p do           // Pour chaque motif de positions
    for each élément pos dans la pile p[gpPref] do          // Pour chaque position du motif
        for each élément gpSuf dans la pile v[pos+lgNewMots] do
            if Intersection(tABRSuf(gpPref),tABRPref(gpSuf]) <> VIDE then // Si gpPref a au moins un suffixe identique à un préfixe de
                gpSuf
                Push(Qa[gpSuf], (gpPref,pos))                                // Ajouter le couple (gpPref,pos) à la pile Qa[gpSuf]
            end if
        end for
    end for
end for
end for

```

Remarques :

1. Dual() est une fonction qui permet de construire un vecteur P à partir d'un vecteur V et inversement. Elle est très simple, je ne la décrirai donc pas ;
2. Push() est une fonction qui permet d'ajouter un élément dans un pile (empiler) ;
3. Intersection() ;
4. Le quorum est toujours vérifié pour les nouvelles extensions.

Algorithm 5 BuildQb : Construction de la pile Qb à partir de la pile Qa et selon les relations pavés.

Arguments d'entrée :

lgNewMots : longueur des motifs en train d'être construits.

Qa : cf. algorithme 4

séquence

Argument de sortie :

Qb : vecteur de pile contenant à la fois la position et un nouveau numéro de motif; il y a au maximum autant de piles dans Qb que de relations possibles (les piles sont indexées sur les numéros de relations). tPS : tableau de correspondance entre les nouveaux numéros de motif et les numéros de préfixe et de suffixe (les numéros de préfixes étaient stockés dans Qa et les numéros de suffixes étaient les indices des piles de Qa). C'est donc un tableau contenant deux éléments : suf, le numéro de suffixe et pref le numéro de préfixe.

Variable locale :

nGpQa : Nombre de motifs dans Qa permettant aussi de re-numéroter les motifs.

pos : position dans la séquence.

gpSuf et gpPref : numéro de motif des suffixes et préfixes.

prev : numéro de motif du couple précédent (pos, gpPref).

pileTemp : pile temporaire contenant toutes les positions d'un motif.

```

nGpQa ← 0
for each pile qui a pour index gpSuf dans Qa do
    prev ← -1
    for each éléments (pos, gpPref) dans la pile Qa[gpSuf] do
        if group = prev then
            prev ← group
        else if group <> prev then
            if CheckQuorum(PileTemp) = Vrai then
                LoadInQb(pileTemp, Qb, nGpQa, lgNewMot, séquence)
                tPS[nGpQa].pref ← gpPref
                tPS[nGpQa].suf ← gpSuf
                nGpQa ← nGpQa + 1
            end if
            prev ← group
            ReInit(pileTemp)
        end if
        Push(pileTemp, pos)
    end for
    if CheckQuorum(PileTemp) = Vrai then
        LoadInQb(pileTemp, Qb, nGpQa)
        tPS[nGpQa].pref ← gpPref
        tPS[nGpQa].suf ← gpSuf
        nGpQa ← nGpQa + 1
    end if
end for

```

sentinelle de changement de pile

// Si si on a changé de pile

sentinelle pour voir si on a changé de pile

// Si on a changé de motif

// Si le quorum est vérifié

// Ventiler les positions du motif dans Qb selon les relations

// Conserver le numéro de préfixe de ce motif

// Conserver le numéro de suffixe de ce motif

// Incrémenter le nombre de motifs

// Ajouter la position pos

// Traiter le dernier motif de la pile

// Si le quorum est vérifié

// Ventiler les positions du motif dans Qb selon les relations

// Conserver le numéro de préfixe de ce motif

// Conserver le numéro de suffixe de ce motif

// Incrémenter le nombre de motifs

Remarques :

- 1) Pour Dual() et CheckQuorum() cf. algorithme 4.
- 2) CheckQuorum() est une fonction qui permet de vérifier le quorum. Par exemple on veut qu'un mot soit présent sur toutes les protéines, cette fonction reverra faux pour tous les motifs pour lesquels ce n'est pas le cas.
- 2) ReInit() est une fonction qui permet de remettre une pile à 0, de vider une pile.
- 2) LoadInQb(pileTemp, Qb, nGpQa, lgNewMot, séquence) est une fonction qui ventile les positions pos contenues dans pileTemp (toujours associée au numéro de motif nGpQa) dans Qb selon les relations entre les symboles en positions pos et pos+lgMot de la séquence. Cette fonction est dépendante des données. 2) Le quorum n'est plus toujours vérifié dans les nouveaux motifs car il n'est pas vérifié lors de la ventilation (ce pourrait être fait mais les opérations à effectuer sont plus compliquées).

Algorithm 6 BuildV : Dépiler Qb dans le vecteurV tout en vérifiant le quorum.

Arguments d'entrée :

Qb : cf. algorithme 5.

Argument de sortie :

V

tCorresGp : tableau de correspondance entre les nouveaux numéros de motifs et les numéros de motifs donnés par BuidQb sur lesquels tPrefSuf est indexé). Il y a de nouveaux numéros car le quorum est vérifié et le nombre de motif diminue.

Variable locale :

nGpQb : Nombre de motifs dans Qb vérifiant le quorum, variable permettant aussi de re-numéroter les motifs

pos : position dans la séquence

nGpQa : numéro de motif attribué par BuildQa et stocké avec les positions dans Qb

iR : indice des piles dans Qb (correspond donc à un numéro de relation)

prev : numéro de motif du couple précédent (pos, gpPref)

pileTemp : pile temporaire contenant toutes les positions d'un motif

```

nGpQb ← 0
for each pile qui a pour index iR dans Qa do                               sentinelle de changement de pile
    prev=-1
    for each éléments (pos, nGpQa) dans la pile Qb[iR] do
        if group = prev then                                              // Si si on a changé de pile
            prev ← group
        else if group<> prev then                                         // Si si on a changé de motif
            if CheckQorum(PileTemp) = Vrai then                                // Si le quorum est vérifié
                LoadInV(pileTemp,V, nGpQb)                                     // Ajouter le numéro de motif nGpQb dansV à toutes les positions de pileTemp
                tCorresGp ← prev
                nGpQb ← nGpQb + 1                                           // Conserver le numéro de préfixe de ce motif
            end if                                                       // Incrémenter le nombre de motifs
            prev ← group
            ReInit(pileTemp)
        end if
        Push(pileTemp, pos)                                                 // Ajouter la position pos
    end for                                                               // Traiter le dernier motif de la pile
    if CheckQorum(PileTemp) = Vrai then                                // Si le quorum est vérifié
        LoadInV(pileTemp,V, nGpQb)                                     // Ajouter le numéro de motif nGpQb dansV à toutes les positions de pileTemp
        tCorresGp ← group
        nGpQb ← nGpQb + 1                                           // Conserver le numéro de préfixe de ce motif
    end if                                                       // Incrémenter le nombre de motifs
end for

```

Remarques :

1) Pour Dual() et Push() cf. algorithme 4.1) Pour CheckQorum(), et ReInit() cf. algorithme 5.2) LoadInV(pileTemp, V, nGpQb) est une fonction qui ajoute le numéro de motif nGpQb à toutes piles de V ayant pour indice les positions dans pileTemp 2) Le quorum est vérifié.

Algorithm 7 FilterV : éliminer les motifs « non maximaux », c'est-à-dire inclus dans d'autres.

Arguments d'entrée :

v : cf. algorithme 2, non filtré

Argument de sortie :

v : cf. algorithme2, filtré

tABRPref : cf. algorithme 2 ; Les ABR sont vides au départ de cette fonction.

tABRSuf : cf. algorithme 2 ; Les ABR sont vides au départ de cette fonction.

Variable locale :

p : vecteur de piles indexé sur les numéros de motif, contenant donc pour chaque pile les positions faisant partie de ce motif.

keep et reject : deux tableaux de booléens initialisés à Faux. groupo et groupi : numéros des motifs testés

iPos : indice dans v

iGp : indice dans p

nGpVf : nombre de motifs après filtre, utilisé aussi pour renommer les motifs.

tABRPrefTmp : cf. algorithme 2 ; Temporaires car dans la première partie, ils sont indexés sur les anciens numéros de motifs (*i.e.* y compris les numéros des motifs non maximaux)

tABRSufTmp : cf. algorithme 2 ; Les ABR sont vides au départ de cette fonction.

```

p=Dual(v)
// Première partie : chercher les motifs « maximaux » (inclus dans aucun autre)
for each pile d'indice iPos dans v do
    for each élément groupo dans la pile v[iPos] do
        if reject[groupo] = Faux then
            for each élément groupi EN DESSOUS de groupo dans v[iPos] do
                if reject[groupi] = Faux then
                    // Si groupo n'est inclu dans aucun des motifs vus jusqu'à présent
                    // Ceux au dessus ont déjà été vus.
                    // Regarder si groupo est inclu dans groupi
                    if keep[groupo] = Faux then
                        // Si groupo n'a pas déjà été déterminé comme maximal
                        if IsInclude(groupo, groupi) =Vrai then
                            // Si groupo est inclu dans groupi
                            // groupo est rejeté
                            FusionABR(groupo, groupi, tABRPrefTmp, tABRSufTmp, tPS, tCorresGp) La liste des préfixes et suffixes de groupo
                            est mise dans celle de groupi
                            break
                        end if
                    end if
                    // Regarder si groupo est inclu dans groupi
                    if keep[groupi] = Faux then
                        // Si groupi n'a pas déjà été déterminé comme maximal
                        if IsInclude(groupi, groupo) =Vrai then
                            // Si groupi est inclu dans groupo
                            // groupi est rejeté
                            FusionABR(groupi, groupo, tABRPrefTmp, tABRSufTmp, tPS, tCorresGp) // La liste des préfixes et suffixes de
                            groupi est mise dans celle de groupo
                        end if
                    end if
                end if
            end for
            keep[groupo] ← ! reject[groupo]
        end if
    end for
end for
for each pile d'indice iPos dans v do
    Reinit(v[iPos])
end for
// Construire le vecteurP filtré
for each pile d'indice iGp dans p do
    if keep[iGp] n Vrai then
        // Pour l'instant, les ABR ne contiennent les que préfixes suffixes des motifs éliminés
        InsertionABR(tPS[tCorresGp[iGp].pref, tABRPrefTmp[iGp]]) On ajoute dans la liste des préfixes le prefixe du motif lui même
        InsertionABR(tPS[tCorresGp[iGp].suf, tABRSufTmp[iGp]]) On ajoute dans la liste des suffixes le suffixe du motif lui même
    for each élément pos dans p[iGp] do
        Push(v[pos], nGpVf)
    end for
    tABRPref[nGpVf] ← tABRPrefTmp[iGp]
    tABRSuf[nGpVf] ← tABRSufTmp[iGp]
    nGpVf ← nGpVf +1
end if
end for

```


ANNEXE B

Formules chimiques des vingt acides aminés

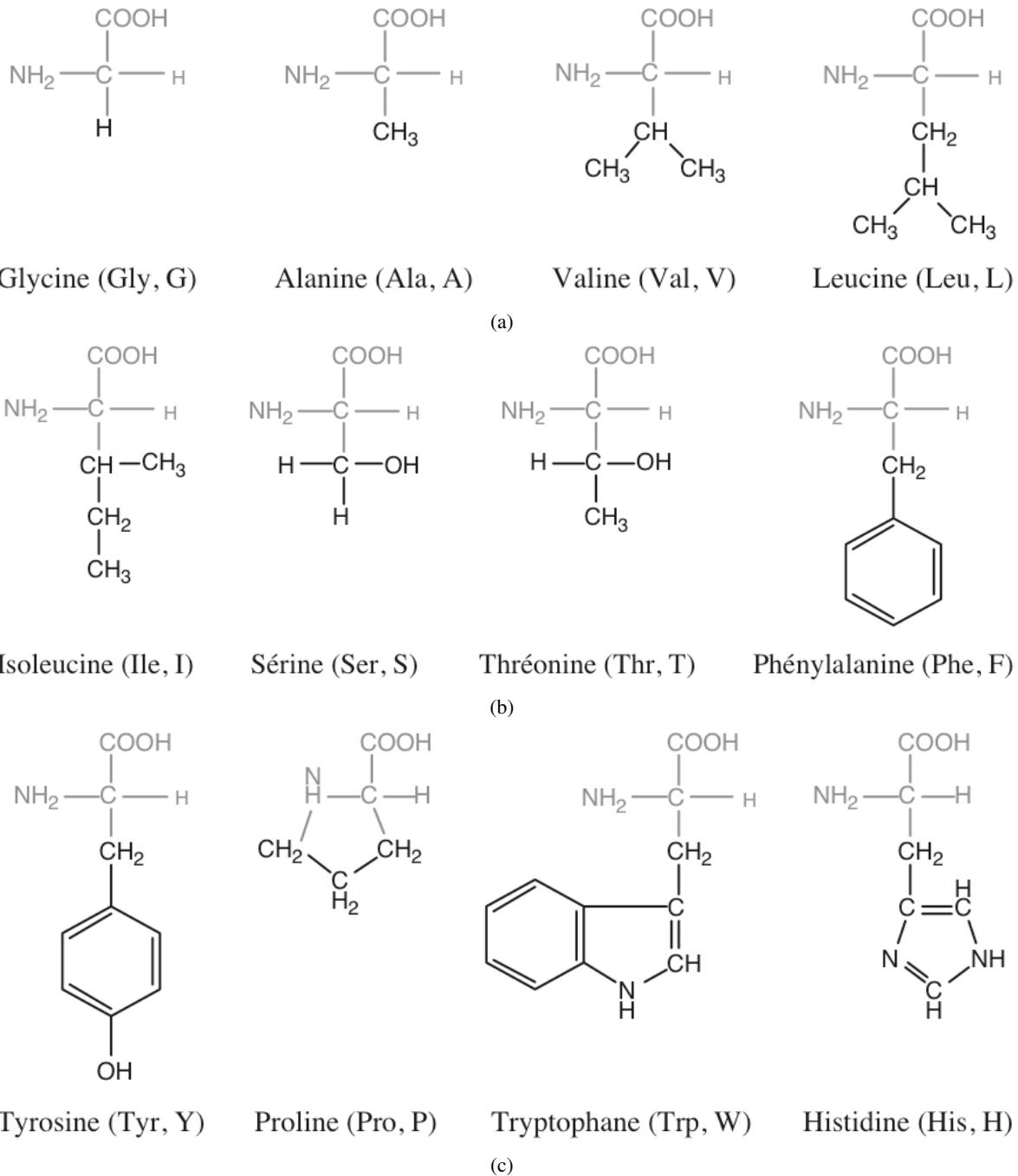


FIG. B.1 – Formules chimiques des 20 acides aminés.

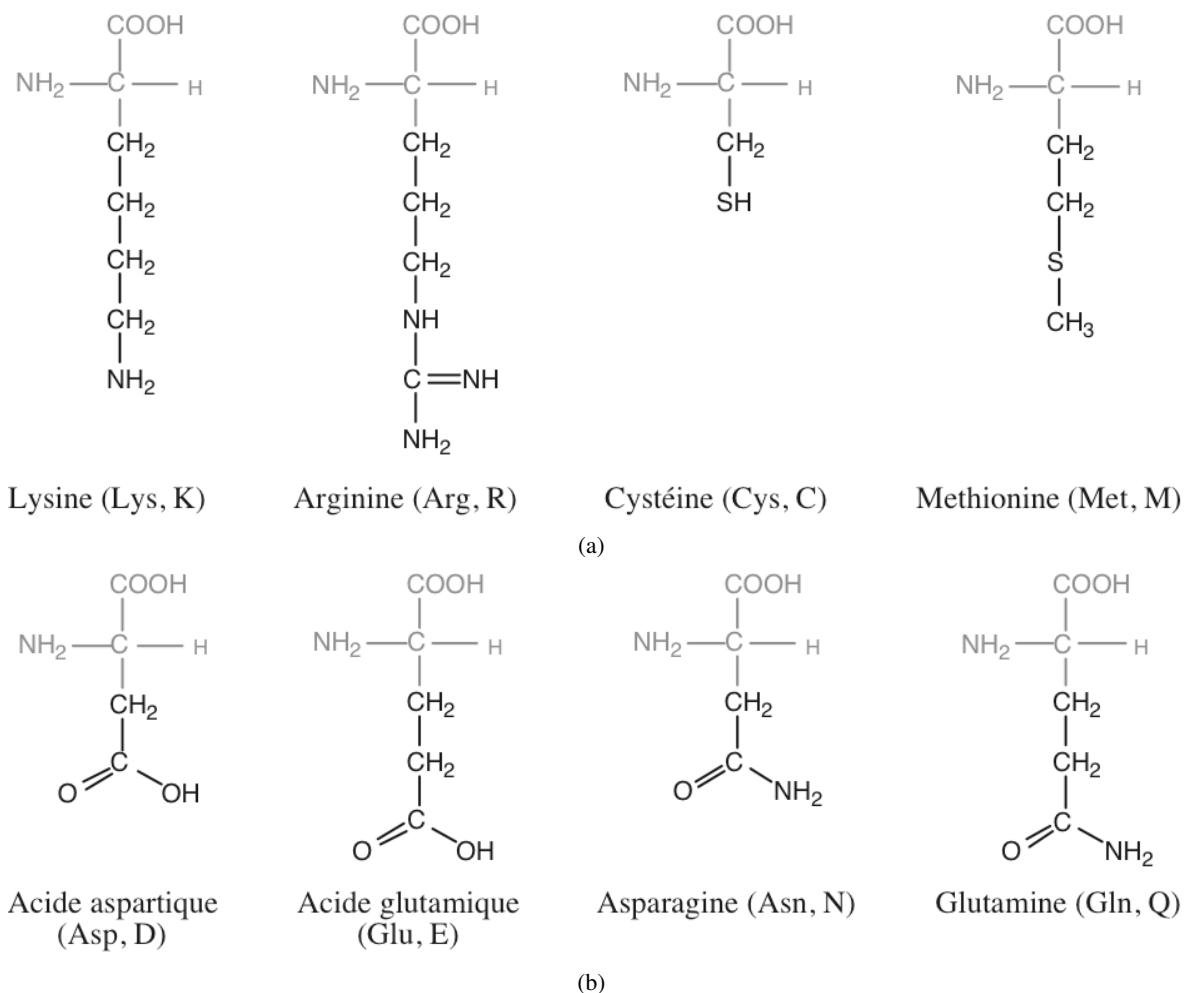


FIG. B.2 – Suite de la figure B.1

ANNEXE C

Publications – conférences

Publications

YAKUSA : a fast structural databases scanning method ;

M. Carpentier, S. Brouillet, J. Pothier ;

Proteins : Structure, Function, and Bioinformatics, Volume 61, *issue* 1, pages 137-51.

RPBS : a web resource for structural bioinformatics ;

C. Allard, F. Moreews, D. Boens, M. Carpentier, S. Chiusa, M. Lonquety, N. Renault, Y. Wong, H. Cantalloube, J. Chomilier, J. Hochez, J. Pothier, B.O. Villoutreix, J.-F. Zagury, P. Tuffery, ; Nucleic Acid Research, Volume 33, *Web Server Issue*, pages W44-9.

Incremental Inference of Relational Motifs with a Degenerate Alphabet ;

N. Pisanti, H. Soldano, M. Carpentier ;

Lecture Note in Computer Science (proceedings CPM, Combinatorial Pattern Matching, Volume 3537, mai 2005, pages 229 - 240.

Implicit and Explicit Representation of Approximated Motifs. ;

N. Pisanti, H. Soldano, M. Carpentier, et J. Pothier ;

dans *Algorithms for Bioinformatics*, éditeurs C. Iliopoulos and K. Park and K. Steinhofel, King's College London Press. (à paraître en 2005).

Inference of Approximated Motifs with Conserved Relations ;

N. Pisanti, H. Soldano, M. Carpentier et J. Pothier ;

Journal of Discrete Algorithms, soumis.

Séminaires / Colloques /Conférences

JOBIM : Journées Ouvertes Biologie Informatique Mathématique, en 2005, Lyon

Alignement multiple de familles protéiques,

M. Carpentier, S. Brouillet, J. Pothier ; poster.

Détection de duplications internes dans les structures des protéines,

A.L. Abraham, M. Carpentier, J. Pothier ; poster.

GGMM : Groupe de Graphisme et de Modélisation Moléculaire, en 2005, l'Île des Embiez, Var

Alignement multiple et local de structures protéiques,

M. Carpentier, S. Brouillet, J. Pothier ; poster.

JOBIM en 2004, Montréal, Canada

Alignement multiple et local de structures protéiques,

M. Carpentier, S. Brouillet, J. Pothier ; communication orale et poster.

ECCB : European Conference on Computational Biology en 2004 à Paris

Structural search in databases ,

M. Carpentier, J. Pothier ; communication orale et poster.

JOBIM en 2002, St Malo

Recherche de similarités structurales dans une banque : YAKUSA,

M. Carpentier, M. Boccaro, J. Pothier ; poster.

Index des méthodes de comparaison structurale décrites

- 3D-Hit, 35
ALIGN, 37
CE, 45
CE-MC, 71
COMPARER, 51, 70
COMPSUP, 70
CONGENEAL, 48
DALI, 43
DEJAVU, 58
ESCAN, 28, 74
Flexprot, 41
FOLDMINER, 59
GRATH, 56
HOMOLOGY, 34
K2/Kenobi, 61
KMRC, 76
LGA, 36
LOCK, 59
MALECON, 71
MAMMOTH, 38, 71
MASS, 76
MATRAS, 60, 72
MaxSub, 34
MINAREA, 37
MNYFIT, 70
MUSAT, 74
POSA, 73
POSSUM, 55
PRIDE, 48
PrISM, 59
Prosup, 36, 37
PROTEP, 56
PSI, 57
RIGOR, 30
SA-Search, 64
SAL, 36
SARF, 56
SGM, 48
SHEBA, 37
SPASM, 29
SSAP, 42, 71
SSM, 56
STAMP, 71
Structal, 36
SuMo, 27
TMAAlign, 37
TOP, 58
TOPSCAN, 63
VAST, 56
WHATIF, 40

Résumé

L'identification des similarités structurales dans les protéines apporte de multiples informations à propos des relations entre les séquences, les structures et les fonctions. Une méthode de recherche de similarités structurales locales dans une banque de structures a été développée. Elle a permis d'établir une procédure de classification des structures en familles. Les blocs structuraux conservés parmi les structures de même famille peuvent ensuite être définis. Trois méthodes de comparaison multiple et locales de structures ont été mises en place, chacune répondant à des critères différents. Elles ont été utilisées pour déterminer des blocs structuraux locaux communs aux protéines d'une même famille.

Ces blocs structuraux communs constitueront une base pertinente de « coeurs » destinée à être intégrée dans un logiciel de reconnaissance de repliement (« threading »). Cette méthode de prédiction des structures protéiques uniquement à partir de leurs séquences peut être utilisée à des fins d'annotation de séquences inconnues, les similarités identifiées pouvant être indécelables par les méthodes utilisant seulement l'information de séquence. L'analyse de ces blocs structuraux apportera aussi de nombreuses informations quant à la conservation de la structure des protéines.

