# Similarités et divergences, globales et locales, entre structures protéiques

Mathilde Le Boudic-Jamin

2015

**THÈSE / UNIVERSITÉ DE RENNES 1**
*sous le sceau de l'Université Européenne de Bretagne*

pour le grade de

**DOCTEUR DE L'UNIVERSITÉ DE RENNES 1**

*Mention : Informatique*

**Ecole doctorale MATISSE**

présentée par


# Mathilde LE BOUDIC-JAMIN

Préparée à l'unité de recherche IRISA, UMR-6074
Institut de recherche en informatique et systèmes aléatoires
ISTIC


---

# Similarités et divergences, globales et locales, entre structures protéiques

**Thèse soutenue à Rennes le 14 décembre 2015**

devant le jury composé de :

**Frédéric CAZALS**
Directeur de recherche INRIA, Sophia Antipolis – Méditerranée / *rapporteur*

**Gurvan MICHEL**
Directeur de recherche CNRS Station Biologique de Roscoff / *rapporteur*

**Annie FORET**
Maître de conférence ISTIC, Université de Rennes 1 / *examinateur*

**Jean-François GIBRAT**
Directeur de recherche INRA, Jouy-en-Josas / *examinateur*

**Jacques NICOLAS**
Directeur de recherche INRIA, Inria Rennes-Bretagne Atlantique / *examinateur*

**Thomas SCHIEX**
Directeur de recherche INRA, Toulouse / *examinateur*

**Rumen ANDONOV**
Professeur à l'Université de Rennes 1 / *directeur de thèse*

Never Give Up, Never Surrender - Galaxy Quest

A Morgan et Aurore.

# Remerciements

L'aventure débuta en 2010, cinq ans de travaux et deux enfants plus tard le voici enfin, le commencement de mes recherches. Cinq années à confronter l'univers de la biologie à celui de l'informatique, cinq années à traduire d'une langue vers l'autre. Je pense que je ne remercierai jamais assez Rumen ANDONOV pour tout ce qu'il m'a apporté, tant en connaissances qu'en méthodologie ou en soutien. Ce fut long, ce fut fatigant, ce fut à la fois destructeur et formateur. Nous avons partagé, confronté, lié nos hypothèses et nos idées, débutant de nombreuses pistes qui méritent toutes d'être explorées. Cette thèse en est le résultat partiel, je n'ai pas pu tout y intégrer tant il y a matière, ce sont de futurs travaux.

Je remercie Frédéric CAZALS et Gurvan MICHEL d'avoir accepté de la rapporter ainsi que les autres membres du jury : Annie FORET, Jacques NICOLAS, Jean-François GIBRAT et Thomas SCHIEX. Je remercie également tous les scientifiques avec qui j'ai collaboré, échangé : Gunnar KLAU, Inken WOHLERS, Jean-François GIBRAT, Frédéric CAZALS, Noël MALOD-DOGNIN, Gurvan MICHEL, Mirjam CZJZEK, Cendrine MONY, Anne-Kristel BITTEBIERE, Jacques NICOLAS, François COSTE, Clovis GALIEZ, Gaelle GARET, Vincent PICARD, Guillaume CHAPUIS, Hristo DJIDJEV, Antonio MUCHERINO, Dominique LAVENIER.

J'ai commencé à Symbiose, alors sous la direction de Jacques, toujours bienveillant (dites Isabelle et Maelle, vous vous en souvenez ?), puis les équipes Genscale, Dyliss et Genouest. Trois équipes, une équipe, la frontière est faible tant les échanges sont réguliers. Je ne pourrai jamais tous les citer, les permanents jamais avares d'un conseil ou d'une petite blague, Dominique, Anne, Olivier, Catherine, Pierre, Olivier, Olivier, Anthony, Fabrice, Claire, François, Nathalie, et j'en oublie déjà, ne m'en veuillez pas. Non François je ne t'ai pas oublié, je tenais à te remercier particulièrement pour toutes nos discussions protéiques, séquence, structure, plus le temps passe et moins l'un va sans l'autre. Et que dire de tous les autres, de Charles qui m'a happée en bioinformatique à Gaelle qui fut toujours là. J'ai une anecdote, un bon moment partagé avec chacun d'entre vous. Initialement il y avait Geoffroy, Sylvain qui pense toujours à m'envoyer un exemplaire d'Harry Potter venant du pays qu'il a visité, grâce à lui et quelques autres je l'ai en huit langues. Nicolas, Nicolas, Sylvain, Thomas, Anaïs, Coline, Laurent, Aymeric, Marie, Chloé, Clovis, Cyril, Renaud, Ivaylo, Rodrigo, Gaétan, Malfoy, Guillaume, Guillaume, Guillaume, Sébastien, Anthony, Marie, Jeanne, merci pour ces moments de détente, d'écoute et tout simplement de vie. Vincent, collègue et ami avec qui j'ai traversé vents et tempêtes, une pensée spéciale pour toi qui rédiges également ta thèse en cet instant. Never Give Up ! Never Surrender ! Gaelle, co-bureau mais tellement

plus. Claudia, co-bureau également, amie, ta douceur innée et ton organisation impeccable m'impressionnent énormément, de plus tu es toujours de bonne compagnie, merci les filles pour tous ces moments à m'écouter râler pour un oui, pour un non. Julie, à toutes nos sorties. Jean, Victorien, ne changez rien. N'oublions pas non plus les déjeuners au Fuji avec Vincent, Guillaume, Charles, Gaelle et Julie, ces moments de détente autour d'un bentô, bientôt une tradition.

Un remerciement tout particulier pour Bruno, Thierry (littéralement sorti d'un placard lors de notre première rencontre), Jean-Marc, Jérôme et Philippe, j'ai maltraité mes machines et ils ont sauvé ma thèse plus d'une fois. Mille merci. Un petit mot aussi pour la plateforme Genouest, j'ai effectué l'équivalent d'une dizaine d'années de calculs sur le genocluster, sans ces ressources et l'aide non négligeable de ceux qui le gèrent mes travaux n'auraient pu aboutir.

Je remercie chaleureusement mes parents, ma famille et ma belle famille. J'ai la chance d'avoir un compagnon qui a grandement sacrifié depuis de nombreux mois pour me permettre de travailler sur cette thèse dans de bonnes conditions. De cela merci Erwan, à charge de revanche mon amour. Oriane, Pierre-Even, une fratrie grandement hétérogène, les outils de comparaison s'y casseraient les dents. Et n'oublions pas les amis, Cécile, ma binôme depuis la L1, Tango Tango Charlie 32 à Chacal dans la nuit, tu me reçois ? Une pensée également pour Thalie, Claire, Pierrick, Manon, Antoine et tous ceux que je n'ai pas cités par manque de mémoire.

Merci à tous.

# Table des matières

# Table des figures

# Liste des tableaux

# Introduction Générale

The protein universe is the set of all proteins of all organisms (Michael Levitt, 2009 [71]). Ce parallèle entre l'infiniment petit et l'infiniment grand illustre bien l'ampleur de la tâche à laquelle s'attellent les scientifiques lorsqu'ils étudient les protéines. Le grand nombre de molécules et le regroupement de certaines via des critères biochimiques (tels les astres regroupés en planètes, étoiles, astéroïdes ou encore météoroïdes) accentuent cette transposition de l'univers chez les protéines. De même, les protéines sont étudiées singulièrement, mais également les unes avec les autres. Elles sont référencées, cartographiées, classifiées, explorées et de même que l'on découvre régulièrement un nouvel astre, de nouvelles protéines apparaissent.

Cette thèse, dédiée aux protéines et plus précisément à leur forme tridimensionnelle, est divisé en trois parties.

La première partie décrit des notions fondamentales en biologie des protéines (chapitre 1) et en théorie des graphes (chapitre B). Un point important développé ici a été de montrer avec précision les atomes des acides aminés qui sont liés spécifiquement à l'une des propriétés de l'acide aminé et d'expliquer comment ces atomes "créaient" cette propriété. S'y trouve également une présentation des données sur lesquelles s'appliquent nos méthodes, que ce soient les domaines structuraux sur lesquels sont appliqués les protocoles d'assignation ou bien la famille d'enzymes (GH5) étudiée en seconde partie. On y trouve également une discussion sur la relation entre la structure des protéines, cœur de notre étude, et la fonction des protéines. Nous nous sommes également intéressés à la modularité des protéines, les changements structuraux issus de milliers d'années d'évolution. Certaines notions concernant les protéines ne sont pas encore explicitement et unanimement définies. Nous avons donc sélectionné certaines définitions qui se retrouvent dans les parties suivantes comme celle du **domaine**.

La seconde partie est dédiée à la classification supervisée des domaines protéiques au sein de bases de données hiérarchiques. La question est ici : comment assigner un domaine structural au sein d'une super-famille d'une classification hiérarchique donnée ? Elle est connue sous le nom de **problème d'identification des familles** (FIP) et nous l'avons abordée avec différents protocoles basés sur une méthode exacte. Ils assurent que la solution trouvée est optimale selon notre score contrairement aux scores d'approximation classiques. Le premier chapitre de cette partie (chapitre 2) introduit les classifications hiérarchiques de domaines structuraux et pose le problème ouvert de la caractérisation de la similarité de deux structures au sein d'une seule et unique valeur : le **score**. Ce score, associé à un alignement de pro-

téines, est souvent utilisé seul pour caractériser une similarité entre deux structures. Il existe pratiquement autant de scores que d'outils de comparaison de structures, par conséquent nous avons créé Samourai, un outil de mesure de scores à partir d'un alignement de deux structures. Pour résoudre le FIP nous avons utilisé le **recouvrement de cartes de contacts** (CMO), une mesure que nous avons utilisée dans toute cette partie. CMO mesure la similarité entre deux structures 3D à partir d'une réduction en 2D de ces structures. Il dénombre un ensemble de similarités locales le long de la globalité des structures. À partir de cette mesure, nous avons utilisé et créé des scores dont l'un possède une propriété importante pour nous : c'est une métrique. Enfin, le chapitre 3 décrit nos différents protocoles, les résultats obtenus et les perspectives d'amélioration.

La troisième partie de cette thèse porte non plus sur la comparaison à grande échelle des structures protéiques, mais au contraire sur la pertinence de la comparaison d'une seule paire de structures détaillée via l'alignement des deux structures. Est-ce que deux structures ont des éléments communs ? Lesquels ? En quelles proportions ? Est-ce que certaines divergences peuvent expliquer une différence de fonction ? La question à laquelle nous avons essayé de répondre est la détection et l'élucidation d'une ou plusieurs similarités locales, et plus importantes : sa signification. Il s'agit d'identifier non seulement la meilleure similarité, mais aussi toutes les "bonnes" similarités. Nous avons cherché à ouvrir le spectre des alignements de structures en ne cherchant pas une zone linéaire dans la séquence correspondant à un bloc bien conservé entre les deux structures, mais au contraire à prendre en compte les différentes modifications liées à l'évolution. Nos expérimentations numériques menées avec des outils d'alignements (chapitre 4), nombreux, ainsi que leurs scores associés, ont montré que toutes les méthodes étaient justifiées. Cela car elle s'orientaient différemment, et qu'elles se spécialisaient dans la détection de similarités différentes. Ainsi, si un outil d'alignement retourne un mauvais alignement, ce n'est pas que les structures ne se ressemblent pas, mais que la similarité cherchée par l'outil n'est pas présente. Par ailleurs, nous montrons que si l'alignement diffère entre deux outils, la superposition peut elle être très similaire et c'est donc l'optimisation, l'extension de l'alignement qui crée une différence. De même s'il n'existe pas un alignement optimal par paire de structures, les alignements alternatifs, bien souvent ignorés par les outils qui ne retournent que le "meilleur" sont une source d'information supplémentaire. Pour effectuer nos recherches, nous avons développé un outil, composé de deux modules complémentaires ayant chacun une tâche bien spécifique :

- Shinobi (chapitre 6) exprime une question biologique dans le formalisme des graphes, il sert à caractériser les protéines et leur comparaison pour faciliter la recherche effectuée du second outil sous des aspects à la fois structuraux et biologiques.
- Ninjas (chapitre 5) est un algorithme de recherche de pseudo-cliques qui parcourt les graphes créés par Shinobi. Ces pseudo-cliques correspondent à de "bons" alignements structuraux et par conséquent, Ninjas recherche indirectement des similarités dans les structures modélisées dans le graphe par Shinobi.

Cet outil nous permet de générer des alignements selon certains critères et ainsi nous aider à comprendre une paire de structures (chapitre 7). Nos avancées sur la détection de similarités locales nous ont menés à nous intéresser aux motifs structuraux répétés dans les protéines. Plus exactement, nous nous sommes intéressés aux motifs qui forment entièrement une

structure, leur détection, leur similarité d'un motif à l'autre. Nous leur avons consacré un chapitre dans lequel nous présentons une méthode de détection supervisée puis *de novo* (implémentée dans l'outil Kunoichi, chapitre 8).

Finalement ,nous avons revisité le problème de la comparaison de structures dans le contexte d'une collaboration avec des chercheurs de la station biologique de Roscoff. Certaines structures sont très semblables, mais leurs fonctions divergent. Nous avons cherché à identifier les zones dans les structures pouvant expliquer ces divergences. L'idée sous-jacente est qu'une légère modification dans la structure ou dans les propriétés physico-chimiques locales directes ou indirectes (c'est-à-dire dans le site catalytique - pour une enzyme par exemple- ou un peu plus loin dans la structure) modifie la fonction. Afin de tester nos hypothèses, nous avons combiné l'étude de la structure et de la biochimie des protéines via un outil nommé Daijinushi (chapitre 9).

Cet outil est un assistant de détection et de visualisation des zones divergentes entre structures similaires, soit l'inverse des méthodes précédentes puisqu'ici on ne cherche pas à mesurer une similarité, mais au contraire à observer des divergences.

Toutes ces expérimentations nous ont permis de fournir des éléments d'étude des protéines, que ce soit de manière globale en nous intéressant aux scores de similarité entre structures ou locale en recherchant toutes les sous-structures similaires ou divergentes d'une structure à l'autre.

# Première partie

# Notions fondamentales

# Chapitre 1

# The Protein Universe

## 1.1 Introduction

Une protéine est, selon sa définition la plus basique, une macromolécule, un ensemble d'atomes interagissant, dans un volume limité. Cette définition est bien en deçà de l'étendue des propriétés des protéines, vague et incomplète, mais, en réduisant à l'extrême, une protéine c'est un mélange de **C**, **N**, **O**, **H** et **S** (pour ne citer que les atomes les plus courants). Toutes les propriétés des protéines découlent des proportions et de la disposition de ces atomes les uns par rapport aux autres. De petites unités structurales, les acides aminés (sous forme de résidus d'acides aminés), se distinguent au sein des protéines et la vision la plus globalement utilisée de la protéine n'est pas l'ensemble d'atomes, mais l'ensemble ordonné de résidus.

Ce chapitre commence par une brève histoire de la science des protéines, puis il est consacré à une présentation des acides aminés protéinogènes que l'on retrouve chez l'Homme. Nous nous sommes attachés à décrire les atomes de ces acides aminés qui portent une propriété spécifique (tels que décrit par Schmitt et confrères dans [105]). Ce, car ces groupes d'atomes ont servi de base à l'un de nos modèles de représentation des protéines (chapitre 6). Ensuite seront présentées les protéines dans leur généralité via quelques rappels sur leurs structures (I, II, III, IV) en développant la structure III qui sera le cœur de ce mémoire. Enfin, nous développerons les relations qui peuvent exister entre ces structures et la fonction des protéines tout en montrant un panel de l'état de l'art et des avancées dans le domaine de l'étude de la fonction des protéines.

## 1.2 Une brève histoire des protéines

Jeannine Yon-Kahn a consacré un livre [121] aux protéines et à leur histoire dans lequel elle retrace tous les travaux et découvertes qui ont mené à l'état de l'art actuel. Cette section en extrait quelques grandes dates et découvertes pour situer le contexte et l'histoire chargée des protéines en espérant montrer que, malgré le nombre de découvertes et d'avancées, l'étude (particulièrement structurale et fonctionnelle) des protéines est loin d'être un sujet clos.

Les premières ébauches d'études des protéines débutent par une macro-observation de leurs effets avec notamment les travaux de Spallanzani qui, en 1783, s'intéressait aux sucs gastriques. Sucs qui font effet grâce à des protéines nommées enzymes. Indirectement, et sans le savoir, Spallanzani étudiait des actions enzymatiques [121]. L'une des premières enzymes (nommées ferments à l'époque) découvertes fut l'amylase, une enzyme salivaire, qui a pour rôle d'hydrolyser l'amidon de malt d'orge. Autrement dit, cette enzyme lyse l'amidon, un grand polysaccharide, en unités plus petites. Ce processus central dans la fabrication de la bière fut découvert par A. Payen et J.F. Persoz en 1833. Le terme enzyme (définition A.20) fut introduit par Kühne en 1878 alors qu'il étudiait le levain. Le $XIX^e$ siècle fut ponctué de découvertes tant sur le principe des enzymes (observation des phénomènes) que sur l'identification des molécules responsables de ces effets et des unités les composant. L'asparagine, l'un des acides aminés (unités des protéines) les plus courants, fut découverte en 1806, et d'autres le furent également tout au long du siècle. En 1931, Vickery et Schmidt publiaient un long article décrivant les acides aminés et synthétisaient tout le travail fait jusqu'alors. Ils ont ainsi posé les bases de domaine de l'étude des acides aminés. Les protéines, sujet central de ce mémoire, furent nommées en 1838 par Berzelius. Leurs compositions et structures sont difficiles à déterminer, car requièrent des étapes d'identification et de purification extrêmement délicates. L'une des premières protéines cristallisées fut l'hémoglobine du ver de terre en 1840 par Hünefeld et la première enzyme, l'uréase par Sumner en 1926.

Si le $XIX^e$ siècle a été le siècle des découvertes des grands principes observés à l'échelle macroscopique, une grande partie du début du $XX^e$ siècle a été celle de grandes avancées scientifiques en chimie avec notamment les travaux de Pauling sur l'électronégativité (définition A.2) en 1932 ou encore les liaisons chimiques. Travaux qui ont entre autres permis de définir les règles principales relatives à la structure des protéines comme la longueur et les angles des liaisons atomiques. La seconde moitié du $XX^e$ siècle a connu des avancées techniques majeures exprimées au travers de la détermination de la structure tridimensionnelle de l'hémoglobine. Les protéines sont des macromolécules difficiles à cristalliser et dont il est encore complexe d'obtenir la structure en trois dimensions, que cela soit avec des techniques de diffraction des rayons X ou par spectroscopie de résonance magnétique nucléaire (RMN). La biologie structurale, l'étude de la structure des protéines est née en 1971 lors d'un symposium intitulé Structure and function of proteins at the three-dimensional level. Par la suite, de plus en plus de structures ont été déterminées via les techniques de diffraction des rayons (cf figure 1.2) tandis qu'en parallèle le nombre de séquences protéiques connues explosait grâce à l'avancée des techniques de séquençage de l'ADN (cf figure 1.1).

Avec l'arrivée de ces données et la popularisation des ressources informatiques sont apparus de nombreux outils d'analyse et de mutualisation de ces données séquentielles, structurales et fonctionnelles. La bio-informatique est le domaine qui centralise traite et aide à l'analyse de ces données ainsi que des résultats et découvertes issus des travaux des biologistes. Les progrès de chacun de ces domaines entraînent des améliorations dans les autres à l'image de l'augmentation de la résolution des structures en cristallographie qui permet d'affiner les seuils des outils de comparaison. En conclusion, l'histoire des protéines est riche et continue à s'enrichir, certains aspects de ce mémoire s'appuient sur des découvertes faites il y a deux cents ans pour tendre à accroître la compréhension de « l'univers des protéines ».

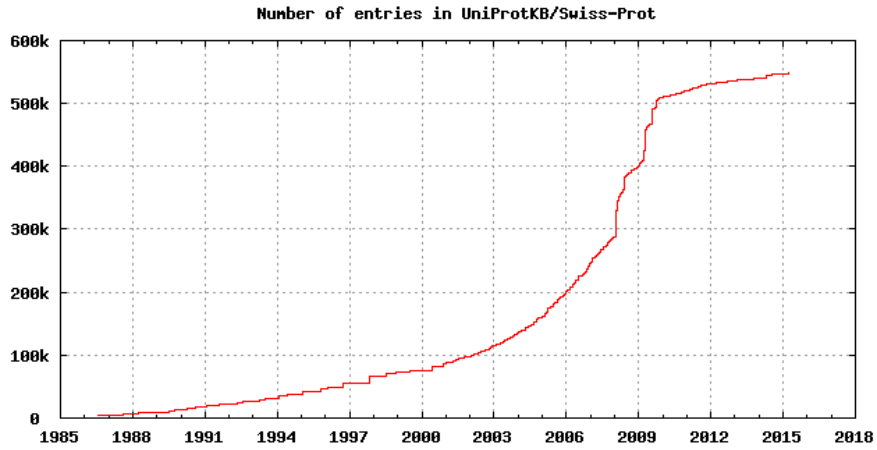



FIGURE 1.1 – Evolution du nombre de séquences protéiques au sein de la base de données de référence UniProtKB-SwissProt

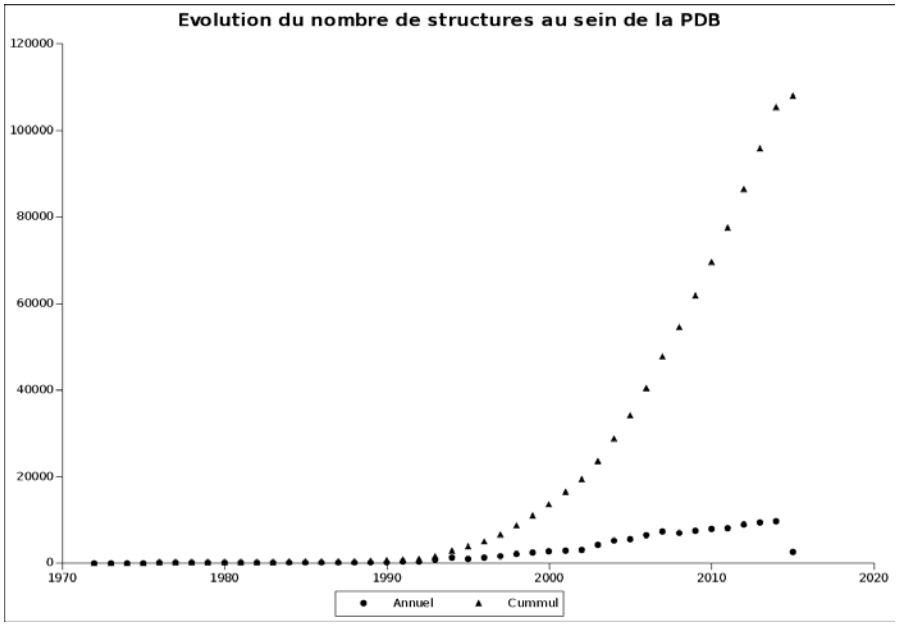



FIGURE 1.2 – Evolution du nombre de structures dans la Protein Data Bank

## 1.3 Les acides aminés, briques du vivant

Un acide aminé (AA) est un acide carboxylique (molécule contenant un groupement **carboxyle**) possédant entre autres un groupement **amine**, d'où son nom. On recense plus de cinq cents acides aminés, cent quarante d'entre eux sont présents chez les protéines, on parle

d'*acides aminés protéinogènes*. Tous possèdent une même partie invariante et ne diffèrent que par leur chaîne latérale (**R**) comme le montre la figure 1.3.

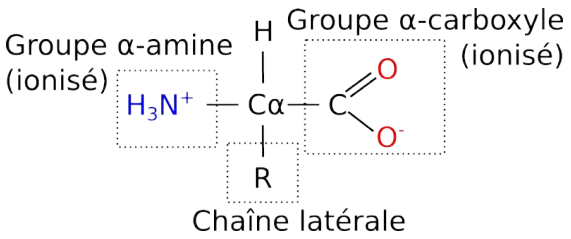



FIGURE 1.3 – Formule générale des acides aminés.

Au centre se trouve un carbone asymétrique, le $C\alpha$, qui est lié au groupement amine, à un hydrogène ainsi qu'un groupement carboxyle et à la chaîne latérale.

### 1.3.1 Acides aminés protéinogènes chez l'Homme

Chez l'Homme il existe vingt et un acides aminés protéinogènes que nous présentons en détail ici. Huit d'entre eux sont dits essentiels, c'est-à-dire que le corps humain ne peut les fabriquer.

- Alanine (ALA, A)
- Arginine (ARG, R)
- Asparagine (ASN, N)
- Acide aspartique (ASP, D)
- Cystéine (CYS, C)
- Glutamine (GLN, Q)
- Acide glutamique (GLU, E)
- Glycine (GLY, G)
- Histidine (HIS, H)
- Isoleucine (ILE, I) - essentiel
- Leucine (LEU, L) - essentiel
- Lysine (LYS, K) - essentiel
- Methionine (MET, M) - essentiel
- Phénylalanine (PHE, F) - essentiel
- Proline (PRO, P)
- Sérine (SER, S)
- Thréonine (THR, T)
- Tryptophane (TRP, W) - essentiel
- Tyrosine (TYR, Y) - essentiel
- Valine (VAL, V) - essentiel
- Sélénocystéine (SEC, U)

Certains contiennent plus d'atomes que d'autres (10 pour la glycine, 27 pour le tryptophane), certains sont très compacts (la leucine par exemple), d'autres plus étendus comme l'arginine ou encore la lysine. Les études de la composition et de la structure des acides aminés (notamment l'observation des différents groupes fonctionnels chimiques -voir annexe A) ont permis de repérer les acides aminés partageant des propriétés/similarités. Par conséquent, on regroupe souvent les acides aminés en fonction de la nature de leurs chaînes latérales et de ces différentes propriétés physico-chimiques qui en découlent.

Le diagramme de Venn suivant, figure 1.4, représente la classification la plus commune des vingt acides aminés les plus courants.

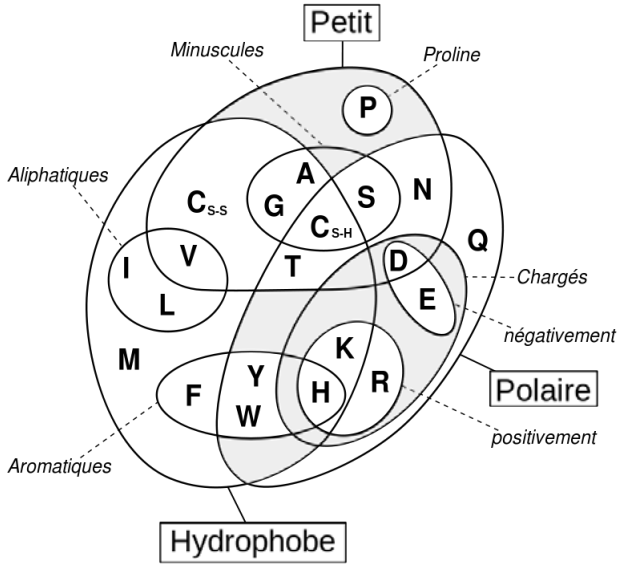



FIGURE 1.4 – Regroupement des acides aminés selon leurs propriétés

Les imbrications des différents groupes montrent un aperçu de l'une des difficultés majeures en biologie moléculaire : expliquer pourquoi une mutation d'un acide aminé par un autre a été maintenue au cours de l'évolution, quels sont les changements induits ou encore peut-on prédire ces changements ? Des études à grande échelle ont montré que les acides aminés apolaires (ou aliphatiques, hydrophobes) sont souvent concentrés à l'intérieur de la protéine ou, lorsqu'ils sont en surface, créent des zones d'accroches/de liaison avec d'autres molécules. À l'inverse, les acides aminés polaires (hydrophiles) se retrouvent en surface, interagissant avec l'eau environnante. D'autres études ont mené à la création de matrices de substitution qui évaluent la probabilité de mutation d'un acide aminé par un autre au sein des protéines.

À titre d'exemple, voici la répartition des différents acides aminés chez l'Homme (figure 1.5), nous calculé à partir du protéome disponible (**UP000005640**) au sein de la base UniprotKB (soit 69 693 séquences protéiques). Sélénocystéine exceptée, nous remarquons que l'acide aminé le plus fréquent (pratiquement 10% pour le leucine) et le moins fréquent (tryptophane, moins de 2%) est tous deux des acides aminés essentiels. Le niveau d'observation est trop élevé pour établir un quelconque résultat marquant, mais cela permet tout de même de montrer que, de par cette non-uniformité des valeurs, la présence d'un acide aminé plutôt qu'un autre n'est pas anodine. Nous allons donc centrer notre étude sur ces vingt acides aminés et leurs propriétés.

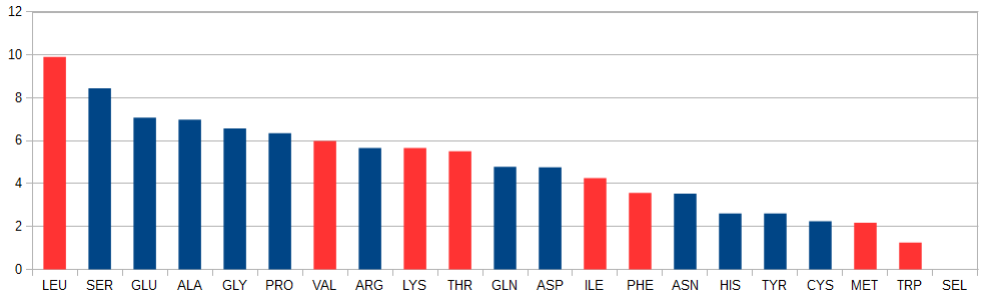



FIGURE 1.5 – Proportion des acides aminés chez l'Homme, en bleu les acides aminés non essentiels, en rouge les acides aminés essentiels

### 1.3.2 Imbriquer des acides aminés et créer des protéines

Les acides aminés s'assemblent en formant des liaisons peptidiques entre les groupes acides carboxyliques **COOH** et les groupes amines **NH**$_3^+$. La chaîne polypeptidique assemblée se compose de résidus d'acides aminés (définition A.19) et constitue la protéine (ou une partie de la protéine). L'assemblage des acides aminés pour former la chaîne polypeptidique se fait au sein du ribosome. L'ARN messager issu de la transcription du gène codant pour la chaîne est traduit en chaîne polypeptidique selon le code génétique (figure 1.6).

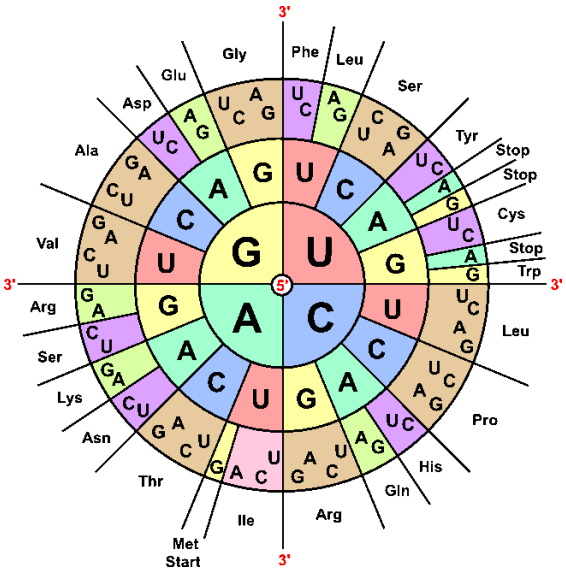



FIGURE 1.6 – Code génétique illustrant la correspondance codon/acide aminé

## 1.4 Groupes fonctionnels de Schmitt *et al.* (FGS)

En 2002, Schmitt *et al.* [105] se sont intéressés aux propriétés des acides aminés et plus exactement aux atomes/groupes d'atomes qui portent ces propriétés. Ces **groupes fonctionnels** (nommés *groupes fonctionnels de Schmitt, ou* ***FGS***, pour les différencier des groupes fonctionnels chimiques) permettent d'associer une propriété globale de l'acide aminé (sa polarité par exemple) à une portion spécifique de l'AA. S'intéresser aux groupes fonctionnels plutôt qu'à l'acide aminé dans sa globalité va permettre d'être plus précis lorsque l'on va comparer deux structures protéiques.

De plus, n'observer une protéine qu'au travers de ses FGS va permettre de s'affranchir des acides aminés. En effet, si un acide aminé porte plusieurs groupes fonctionnels, un groupe fonctionnel est de même présent chez plusieurs acides aminés.

Enfin, les FGS permettent d'expliquer les propriétés des acides aminés. Par exemple pourquoi la lysine est-elle à la fois polaire (hydrophile) et hydrophobe ? L'hydrophilie est due au groupement amine ($N\zeta H_2$) de la chaîne latérale qui va avoir tendance à partager son hydrogène tandis que l'hydrophobie se situe au niveau du groupe aliphatique créé par les quatre carbones ($C\beta$, $C\gamma$ , $C\delta$ et $C\epsilon$) de la chaîne latérale.

Les auteurs décrivent dans leur article [105] cinq types de FGS pour les acides aminés :
- *donneur d'hydrogène* ($DO$)
- *accepteur d'hydrogène* ($AC$)
- *donneur/accepteur d'hydrogène* ($DA$)
- *groupe aliphatique* ($\triangle$)
- *cycle aromatique* ($\circ$)

Chaque acide aminé (sauf la glycine) porte sur sa chaîne latérale un ou plusieurs groupes fonctionnels. Lorsque plusieurs atomes interviennent dans un groupe fonctionnel (typiquement un cycle aromatique), les auteurs ont considéré un pseudo-centre, barycentre des coordonnées de chaque atome impliqué. Nous appliquerons la même méthode. La partie squelettique (soit les atomes des résidus moins la chaîne latérale) est aussi dotée de groupes fonctionnels, l'azote est donneur d'hydrogène, l'oxygène accepteur d'hydrogène et le carbone qui lui est lié a des propriétés aromatiques.

### 1.4.1 Répartition des groupes fonctionnels au sein des acides aminés et explications des propriétés physico-chimiques

Cette section décrit chaque acide aminé, ses groupements fonctionnels et les propriétés qui en découlent. Dans les formules développées, les différents groupes fonctionnels des chaînes latérales sont signalés :

- groupe fonctionnel aromatique : ○
- groupe fonctionnel aliphatique : △
- groupe fonctionnel donneur d'hydrogène : •
- groupe fonctionnel donneur/accepteur d'hydrogène : •
- groupe fonctionnel accepteur d'hydrogène : •

De même, sur les représentations 3D suivantes apparaissent les atomes selon la convention :

- les atomes de carbone (**C**) en gris ;
- les atomes d'oxygène (**O**) en rouge ;
- les atomes d'azote (**N**) en bleu ;
- les atomes de soufre (**S**) en jaune ;

Les atomes d'hydrogène (**H**) ne sont pas représentés dans les représentations 3D afin d'éclaircir l'image, de même le groupe hydroxyle ($OH$) de la partie squelettique de l'acide aminé (ensemble des atomes n'appartenant pas à la chaîne latérale) n'est pas présent, car l'acide aminé est représenté sous sa forme de résidu (soit déchargé d'un oxygène et de deux hydrogènes).

#### Alanine (ALA, A)

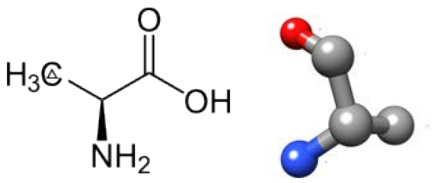



FIGURE 1.7 – Alanine
Formule développée (g) et visualisation 3D (d).

L' alanine ($C_3H_7NO_2$, figure 1.7) est un acide aminé dont la chaîne latérale est composée d'un groupement méthyle (un carbone ($C\beta$) lié à trois hydrogènes et relié au carbone central ($C\alpha$)). Cette chaîne latérale de l'alanine est un petit groupement aliphatique. De fait c'est un petit (minuscule même) acide aminé hydrophobe (le $C\beta$ ne partageant pas ses hydrogènes) et apolaire.

**Arginine (ARG, R)**

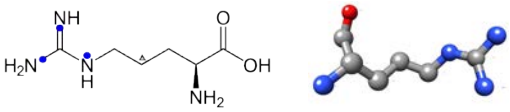



FIGURE 1.8 – Arginine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

L'arginine ($C_3H_{14}N_4O_2$) figure 1.8), est un acide aminé polaire et chargé positivement. Une portion de sa chaîne latérale (les trois premiers carbones -$C\beta C\gamma C\delta$) est un groupement aliphatique tandis que ses trois groupements amines (N$\epsilon$, NH1 et NH2) sont des donneurs d'hydrogène (aka tendent à partager leurs liaisons hydrogène avec de l'eau). La présence de ces groupements amines permet à l'arginine d'interagir avec des acides aminés chargés eux négativement (D,E). Les groupes amines peuvent former des liaisons ioniques avec des groupes hydroxyle (OH), créant ainsi des ponts salins comme l'illustre la figure 1.9. Ces ponts ont pour effet d'aider à maintenir la structure de la protéine à laquelle ils appartiennent.

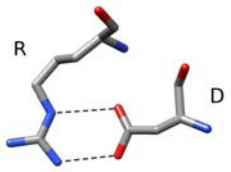



FIGURE 1.9 – Ponts salins entre une arginine (**R**) et un aspartate (**D**)

**Asparagine (ASN, N)**

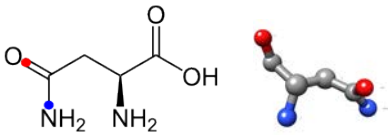



FIGURE 1.10 – Asparagine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

L'asparagine ($C_4H_8N_2O_3$, figure1.10) est un acide aminé hydrophile, polaire non chargé. La chaîne latérale est constituée du $C\beta$ et d'une fonction acide carboxylique ($-C(O)OH$) amidifiée, c'est à dire que l'un des oxygènes a été substitué par $-NH_2$ pour donner la forme

($\breve{}C(NH_2)OH$). Le groupe $-OH$ est accepteur d'hydrogène tandis que le groupe $-NH_2$ est donneur d'hydrogène.

**Acide aspartique (ASP, D)**

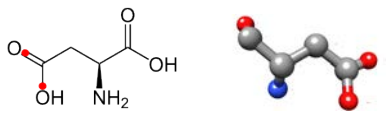



FIGURE 1.11 – Acide aspartique
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

L'acide aspartique $C_4H_7NO_4$ possède une chaîne latérale composée du $C\beta$ auquel s'ajoute un groupement carboxyle dont les oxygènes sont accepteurs d'hydrogène. Ce groupe carboxyle confère une polarité négative à l'acide aminé, les oxygènes étant de forts attracteurs d'électrons. On distingue donc ici deux FGS (un par oxygène). De plus, l'acide aspartique peut former des ponts salins avec des acides aminés chargés positivement (**K**, **H**, **R**) comme le montre la figure 1.9.

**Cystéine (CYS, C)**

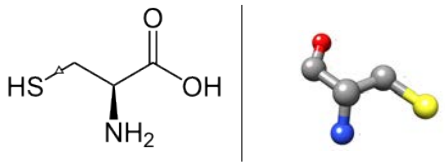



FIGURE 1.12 – Cystéine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

La cystéine ($C_3H_7NO_2S$, figure 1.12) possède un groupement sulfhydryle ou **thiol** (SH) au bout du $C\beta$ qui permet la formation de pont disulfure (figure 1.13). Ces ponts sont les plus stables observables au sein des protéines et contribuent donc au maintien de la forme de la protéine. La chaîne latérale étant saturée, elle forme un petit groupe aliphatique.

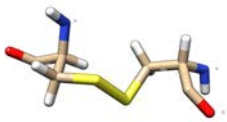


FIGURE 1.13 – Pont disulfure (en jaune) entre deux cystéines chez la protéine 2K73

**Glutamine (GLN, Q)**

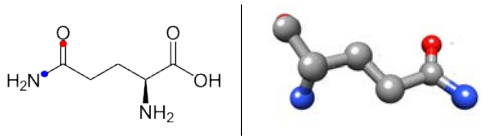



FIGURE 1.14 – Glutamine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

Le glutamine ($C_5H_10N_2O_3$, figure 1.14) est polaire non-chargé et hydrophile grâce au groupe amine ($-NH_2$), donneur d'hydrogène et au groupe ($-OH$), accepteur d'hydrogène qui forment sa chaîne latérale. La chaîne latérale globale est constituée de trois carbones et des groupes carboxyles et amines.

**Glutamate (GLU, E)**

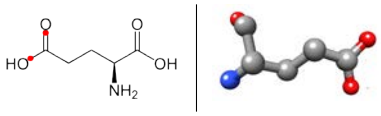



FIGURE 1.15 – Aide glutamique
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

Sa chaîne latérale contient un résidu carboxyle, ce qui en fait un acide aminé « acide », dicarboxylique, polaire. Le glutamate ($C_5H_9NO_4$, figure 1.15) est un résidu acide (de par son groupe carboxyle $HO^--$) très polaire, chargé négativement et hydrophile. Comme l'aspartate, il peut créer des liaisons électrocovalentes avec les charges positives de l'arginine, de la lysine et de l'histidine. Cela aide au maintien de la structure de la protéine, mais peut aussi servir de point d'ancrage lors d'interactions protéine/ligand. Les deux oxygènes ($HO\delta 2^--$ et $0\delta 1$) sont des accepteurs d'hydrogènes et de forts attracteurs.

**Glycine (GLY, G)**

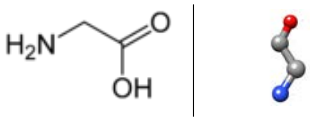



FIGURE 1.16 – Glycine
Formule développée (g) et visualisation 3D (d).

La glycine ($C_2H_5NO_2$, figure 1.16) est un acide aminé particulier puisqu'il est le seul à ne pas posséder de $C\beta$ dans sa chaîne latérale qui n'est composée que d'un hydrogène. Cela fait de la glycine un acide aminé minuscule (seulement dix atomes), apolaire et hydrophobe.

**Histidine (HIS, H)**

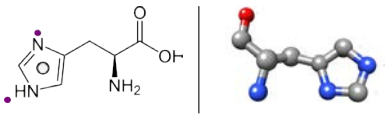



FIGURE 1.17 – Histidine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

L'histidine ($C_6H_9N_3O_2$) est un acide aminé avec une chaîne latérale basique, donc polaire (chargée positivement) de par la présence de deux atomes d'azote (**N**). Le radical est composé du $C\beta$ et d'un *imazole*, ce dernier a, par nature, des propriétés aromatiques et ses deux azotes peuvent partager leurs hydrogènes ou en attirer (ce sont des donneurs accepteurs d'hydrogène, $DA$). Le modèle de Schmitt *et al.* [105] y distingue donc trois groupes fonctionnels : $\circ$, $DA$ et $DA$ (1.17, à g.).

**Isoleucine (ILE, I) - acide aminé essentiel**

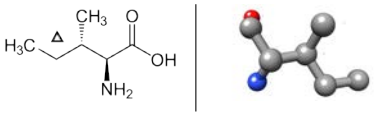



FIGURE 1.18 – Isoleucine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

L'isoleucine ( $C_6H_13NO_2$,figure 1.18) est un acide aminé très hydrophobe et non polaire dont la chaîne latérale (isobutyle) est ramifiée. Elle est composée de quatre carbones dont deux formant des groupes $CH_3$ et un carbone asymétrique ($C\beta$). L'agencement ramifié et compact de la chaîne latérale limite les conformations que peut adopter l'isoleucine, une conséquence de cette caractéristique est que l'isoleucine a du mal à intégrer une formation en hélice$\alpha$, à l'inverse cet acide aminé se retrouve facilement dans des feuillets $\beta$. L'isoleucine est représentée, en terme de groupes fonctionnels par un groupe aliphatique, barycentré entre les quatre carbones de la chaîne latérale ($C\beta$, $C\gamma1$, $C\gamma2$ et $C\delta1$).

**Leucine (LEU, L) - acide aminé essentiel**

La leucine possède la même formule brute que l'isoleucine [1.4.1] ( $C_6H_13NO_2$,figure 1.19) et est également ramifié, non-polaire, avec en chaîne latérale (isobutyle) un $C\beta$ (sy-

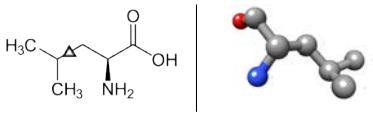



FIGURE 1.19 – Leucine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

métrique ici) suivi d'un $C\gamma$ et deux groupes méthyles $C\delta H_3$. De fait, la leucine possède les mêmes propriétés que l'isoleucine, un groupe aliphatique ($\triangle$) dû à la saturation de la chaîne carbonée. La leucine se retrouve généralement à l'intérieur des protéines solubles ou en contact avec des lipides dans les hélices membranaires.

**Lysine (LYS, K) - acide aminé essentiel**

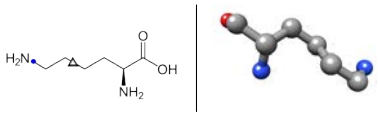



FIGURE 1.20 – Lysine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

La lysine ($C_6H_14N_2O_2$, figure 1.20) est globalement un acide aminé basique (polaire chargé positivement), néanmoins, sa chaîne latérale composée de quatre $CH_2$ forme un groupe aliphatique (hydrophobe) et d'un groupe amine ($NH_3^+$) est amphiphile. De plus, comme l'arginine (**R**) et l'histidine (**H**), la lysine peut former des ponts salins avec des acides aminés chargés négativement (**E**, **D**). Dans certains cas, la lysine peut se substituer à un acide aminé ayant un cycle aromatique (e.g. **H**) et assurer la fonction aromatique avec son groupement aliphatique, c'est ici la propriété hydrophobe qui est conservée lors de la substitution.

**Méthionine (MET, M) - acide aminé essentiel**

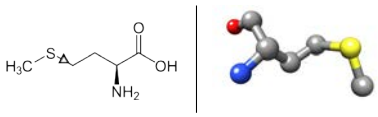



FIGURE 1.21 – Méthionine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

La méthionine ($C_5H_11NO_2S$, figure 1.21) est, comme la cystéine, un acide aminé soufré hydrophobe. L'atome de soufre participe à ce que l'on appelle une fonction thioéther ($--$

$SCH_3$). Au niveau groupe fonctionnel, la chaîne latérale possède un groupe aliphatique ($\triangle$) formé par les quatre atomes lourds ($C\beta$, $C\gamma$, $S\delta$ et $C\epsilon$) et responsable de la propriété hydrophobe de l'acide aminé. La méthionine est peu réactive, mais, grâce à sa propriété hydrophobe, peut servir dans la reconnaissance de sites de liaison ou de reconnaissance de ligands hydrophobes. Quant à l'atome de soufre, n'étant pas en position terminale (il est suivi d'un méthyle), il interagit difficilement avec d'autres atomes extra-moléculaires.

**Phénylalanine (PHE, F) - acide aminé essentiel**


FIGURE 1.22 – Phénylalanine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

La phénylalanine ($C_9H_{11}NO_2$, figure 1.22) est un acide aminé aromatique hydrophobe ; sa chaîne latérale comprend un cycle aromatique monocyclique (**C** uniquement) avec six électrons délocalisés. Ce cycle peut interagir avec les chaînes latérales d'autres acides aminés comportant également un cycle aromatique (**Y**, **H**, **W**). Le radical de la phénylalanine contient uniquement le $C\beta$ suivi du cycle aromatique ($\circ$) composé de six carbones ($C\gamma C\delta 1 C\delta 2$ $C\epsilon 1$ $C\epsilon 2$ $C\zeta$).

**Proline (PRO, P)**

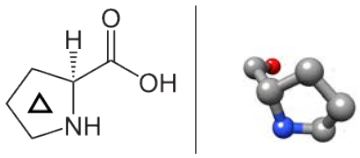



FIGURE 1.23 – Proline
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

Contrairement aux autres acides aminés, la chaîne latérale de la proline ($C_5H_9NO_2$,figure 1.23) n'est pas connectée une, mais deux fois au squelette, la première au $C\alpha$comme les autres et la seconde au groupe N-terminal. Cette double connexion permet à la chaîne latérale de former un anneau composé de cinq membres ($C\alpha$, $C\beta$, $C\gamma$, $C\delta$, $NH_2$) et contraint fortement les conformations que peut adopter l'acide aminé. Les trois carbones de la chaîne latérale sont saturés et forment donc un groupe aliphatique ($\triangle$).

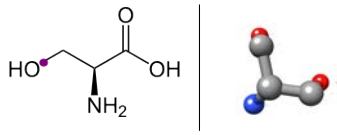



FIGURE 1.24 – Sérine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

### Sérine (SER, S)

La sérine ($C_3H_7NO_3$, figure1.24) est un petit acide aminé polaire de par sa fonction alcool (portée par la chaîne latérale ( $C\beta H_2 - -OH$), faiblement hydrophile et dont l'oxygène du radical est donneur/accepteur d'hydrogène ($DA$)). La sérine est souvent en contact avec le solvant et peut participer aux sites catalytiques des enzymes.

### Threonine (THR, T) - acide aminé essentiel

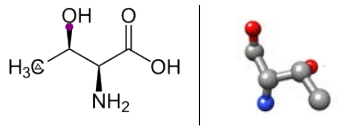



FIGURE 1.25 – Thréonine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

La thréonine ($C_4H_9NO_3$, figure1.25) est un petit acide aminé ramifié, à l'image de la leucine. Son groupe méthyle, $-CH_3$ (saturé donc) est aliphatique ($\triangle$) tandis que l'autre ramification est un groupe hydroxyle ($-OH$) tantôt donneur, tantôt accepteur d'hydrogène ($DA$) suivant l'environnement. Ce groupe $-OH$ définit la propriété polaire de l'acide aminé et, combiné au $C\beta$, forme une fonction alcool.

### Tryptophane (TRP, W) - acide aminé essentiel

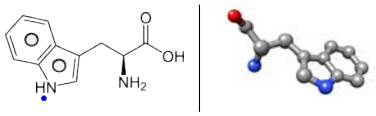



FIGURE 1.26 – Tryptophane
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

Le tryptophane ($C_{11}H_{12}N_2O_2$, figure 1.26) est un gros acide aminé aromatique, amphiphile. Son radical est un indole, c-à-d que son composé aromatique est hétérocyclique ; en

effet il est constitué d'un cycle pyrrole (quatre **C** et un **N**) et d'un cycle benzénique (six **C**) partageant deux atomes de carbone. Ces deux cycles ($\circ$) le rendent hydrophobe, mais le groupe $-N\epsilon H$ est donneur d'hydrogène ($DO$) ce qui permet quelques interactions.

**Tyrosine (TYR, Y)**

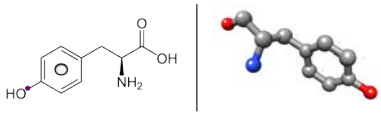



FIGURE 1.27 – Tyrosine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

La tyrosine ($C_9H_{11}NO_3$, figure 1.27) est amphiphile non chargée avec en chaîne latérale un phénol (un cycle aromatique portant un groupe hydroxyle $-OH$). On distingue donc ici deux groupes fonctionnels : le cycle aromatique (hydrophobe) $\circ$ et le groupe hydroxyle donneur/accepteur d'hydrogène ($DA$).

**Valine (VAL, V) - acide aminé essentiel**

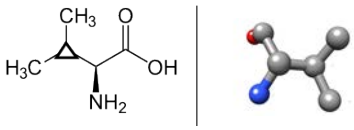



FIGURE 1.28 – Valine
Formule développée de l'acide aminé (g) et visualisation 3D de la forme résiduelle (d).

La valine ($C_5H_{11}NO_2$, figure 1.28) est un acide aminé hydrophobe aliphatique. Sa chaîne latérale, ramifiée, carbonée et saturée, est un isopropyle (le $C\beta$ est lié à deux groupes méthyles ($-CH_3$). Fonctionnellement parlant, la valine n'est constituée que d'un groupe aliphatique ($\triangle$) barycentré entre les carbones du radical ($C\beta$, $C\gamma 1$ et $C\gamma 2$).

### 1.4.2 Résumé du point de vue des groupes fonctionnels (FGS)

Le tableau 1.1 répertorie, pour chaque acide aminé, le ou les groupes fonctionnels présents sur la chaîne latérale. Nous avons observé plus haut (cf figure 1.5 la proportion en acides aminés au sein du protéome humain. Nous avons calculé la proportion en groupes fonctionnels pour les acides aminés essentiels et non essentiels (figure 1.29). Ce graphique permet d'observer que les acides aminés essentiels sont composés à plus de 20% de groupements aliphatiques, porteurs de la propriété hydrophobe des acides aminés. En revanche les acides aminés non essentiels sont plus polarisés, ce qui permet les liaisons avec d'autres molécules.

En résumé le corps humain synthétise les acides aminés portant les groupes fonctionnels "partageurs d'hydrogène", mais les fonctions aliphatiques viennent en majorité d'acides aminés essentiels.

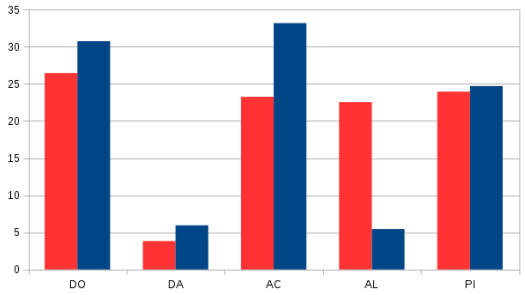



FIGURE 1.29 – Pourcentage en groupes fonctionnels au sein du protéome humain pour les acides aminés essentiels (en rouge) ou non

TABLE 1.1 – Groupes fonctionnels présents chez les acides aminés
Donneur d'**H** : $DO$, Accepteur d'**H** : $AC$, Donneur/Accepteur d'**H** : $DA$, Cycle aromatique : PI, Groupement aliphatique : ALI

| Acide amine | DO | AC | DA | PI | ALI | AA essentiel |
|---|---|---|---|---|---|---|
| Alanine | - | - | - | - | 1 | - |
| Arginine | 3 | - | - | - | 1 | - |
| Asparagine | 1 | 1 | - | - | - | - |
| Acide aspartique | - | 2 | - | - | - | - |
| Cystéine | - | - | - | - | 1 | - |
| Glutamine | 1 | 1 | - | - | - | - |
| Acide glutamique | - | 2 | - | - | - | - |
| Glycine | - | - | - | - | - | - |
| Histidine | - | - | 2 | 1 | - | - |
| Isoleucine | - | - | - | - | 1 | Oui |
| Leucine | - | - | - | - | 1 | Oui |
| Lysine | 1 | - | - | - | 1 | Oui |
| Méthionine | - | - | - | - | 1 | Oui |
| Phénylalanine | - | - | - | 1 | - | Oui |
| Proline | - | - | - | - | 1 | - |
| Sérine | - | - | 1 | - | - | - |
| Thréonine | - | - | 1 | - | 1 | Oui |
| Tryptophane | - | - | 1 | 2 | - | Oui |
| Tyrosine | - | - | 1 | 1 | - | - |
| Valine | - | - | - | - | 1 | Oui |

## 1.5 Protéines

Le terme protéine vient du grec ancien $\pi\rho\omega\tau o\varsigma$ (*protos*) signifiant premier, essentiel. Les protéines, de par l'étendue des fonctions qu'elles assurent dans un organisme se retrouvent partout, certaines ont un rôle structurel comme l'actine, un rôle de transport (hémoglobine) ou encore catalysent des réactions (enzymes), et la liste n'est pas exhaustive. Une protéine est composée d'acides aminés, qui, par leurs propriétés physico-chimiques et leurs interactions, définissent son repliement dans l'espace et sa fonction.

### 1.5.1 Structure primaire (I) : Séquence

La structure primaire d'une protéine est sa composition en résidus d'acides aminés (notés simplement résidus ici). Cette structure est communément appelée la **séquence** d'une protéine.

**Définition 1.1 (Séquence d'une protéine)** *Composition ordonnée en résidus d'acides aminés d'une protéine.*

Ces résidus sont enchaînés les uns à la suite des autres par des liaisons peptidiques (reliant l'extrémité C-terminale d'un acide aminé à l'extrémité N-terminale d'un autre). La séquence d'une protéine se lit de l'extrémité N-terminale vers l'extrémité C-terminale.

### 1.5.2 Structure secondaire (II)

Le concept de structure secondaire a été introduit par Kaj Ulrik Linderstrøm-Lang, lors des conférences médicales Lane en 1952 à Stanford.

La structure secondaire d'une protéine est l'ensemble des structures locales formées par liaisons hydrogènes entre atomes des résidus d'acides aminés. Il existe deux grandes classes de structures secondaires : les hélices et les feuillets (notées respectivement $\mathcal{H}$ et $\mathcal{S}$). Les résidus n'intervenant pas dans les structures secondaires précédentes sont par défaut regroupés dans une troisième super classe (notée $\mathcal{O}$).

La formation d'une structure secondaire est influencée par la nature des acides aminés qui la compose. En effet, certains acides aminés favorisent la formation d'hélices (Alanine, Glutamate, Leucine, Lysine et Méthionine) tandis que d'autres sont plus souvent retrouvés dans des feuillets (Isoleucine, Phénylalanine, Threonine, Tryptophane, Tyrosine et Valine).

Il existe plusieurs algorithmes de prédiction de structures secondaires comme STRIDE [45], KAKSI [77] ou DSSP [68] (ce dernier étant le plus couramment utilisé). DSSP distingue huit classes de structures secondaires présentées ci-dessous (Table 1.2) que nous pouvons regrouper dans les trois super-classes $\mathcal{H}$ $\mathcal{S}$ et $\mathcal{O}$.

### 1.5.3 Structure tertiaire (III)

Lorsque nous parlons communément de la **structure d'une protéine,** c'est en vérité à la *structure tertiaire - ou tridimensionnelle-* (structure III ou structure 3D) qu'il est fait référence (définition 1.2).

TABLE 1.2 – Définition des catégories de structures secondaires selon DSSP

| Superclasse | $\mathcal{H}$ | $\mathcal{S}$ | $\mathcal{O}$ |
|---|---|---|---|
| Classes | **G** : Hélice 310<br>**H** : Hélice $\alpha$<br>**I** : Hélice $\pi$ | **E** : Brin $\beta$ dans un feuillet antiparallèle/parallèle<br>**B** : résidu isolé dans un pont $\beta$ | **T** : Coude (avec liaison hydrogène)<br>**S** : Coude (sans liaison hydrogène)<br>**C** : Boucle (catégorie par défaut) |

**Définition 1.2 (Structure d'une protéine)** *La structure d'une protéine est l'agencement spatial de ses atomes. Cet agencement, ce* ***repliement*** *de la protéine (Fold) est contraint par la composition chimique de la protéine, mais également par des facteurs environnementaux (nature du solvant, température, salinité, PH...).*

La structure d'une protéine est maintenue par différentes interactions entre ses atomes (interactions présentées dans l'annexe A : les interactions covalentes, les interactions électrostatiques, les interactions de Van der Waals ou encore les interactions ioniques... La variété des structures protéiques, le nombre de repliements observés sont bien moindres que le nombre de séquences connues, c'est le paradoxe de Levinthal [127]. Le paradoxe provient du contraste entre le nombre de conformations possibles (considérable même pour de petites protéines) et la durée de repliement effectif des protéines. De ce paradoxe est déduit qu'une protéine ne « teste » pas toutes les conformations possibles lors du repliement mais ce repliement est guidé par des interactions résidu-résidu.

### 1.5.4 Structure quaternaire (IV)

Une protéine se compose d'une ou plusieurs chaînes polypeptidiques. L'agencement de ces chaînes constitue la structure quaternaire de la protéine.

## 1.6 Domaine protéique

Le terme *domaine* protéique a plusieurs définitions selon les domaines [92] et associant une séquence protéique à une fonction (suivant la définition fonctionnelle). Il existe de nombreuses bases de données répertoriant les séquences protéiques correspondant à un domaine tel que CDD [75] (Conserved Domain Database), ProDom [20], ou encore InterPro [55] qui retourne une analyse fonctionnelle d'une séquence protéique en la comparant à des signatures telles que celles de PROSITE [54].

La définition générale d'un domaine du point de vue structural [108] est la suivante :

**Définition 1.3 (Domaine structural)** *Un domaine structural est une région compacte de la chaîne polypeptidique capable de se replier de manière stable et en conservant certaines ou toutes ses capacités lorsqu'extrait de la protéine entière. Une chaîne polypeptidique peut*

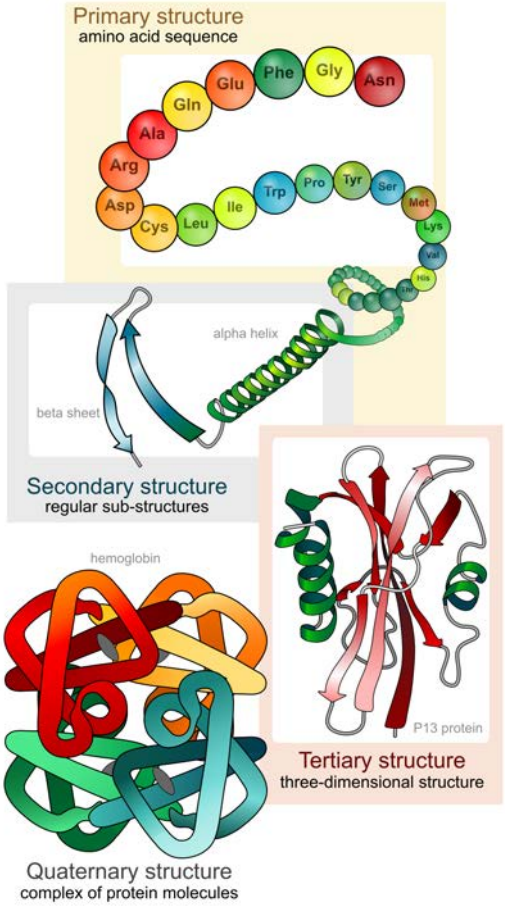



FIGURE 1.30 – Structure I, II, III, IV des protéines

https://upload.wikimedia.org/wikipedia/commons/thumb/c/c9/Main_protein_structure_levels_en.svg/2000px-Main_protein_structure_levels_en.svg.png

*ainsi contenir un ou plusieurs domaines structuraux. Les premiers domaines étaient identifiés manuellement puis des algorithmes de détection automatique sont apparus.*

Un domaine est donc une portion de protéine compacte et stable pouvant porter une fonction. La détermination d'un domaine, du point de vue structural, se fait majoritairement en analysant les interactions entre atomes pour évaluer la compacité de la portion de protéine considérée. Holm et Sander [50] par exemple, ont développé une méthode basée sur des mesures de distances interatomiques et sur les interactions entre atomes pour séparer les domaines. Les domaines sont définis comme des zones compactes qui interagissent peu les unes avec les autres (e.g. la chaîne A de la protéine **1C9B**, figure 1.31). D'autres outils existent comme CATHEDRAL [99].

Un domaine est le plus souvent composé d'une section continue de la chaîne polypeptidique citePetsko2004 mais certains contre-exemples ont été observés comme dans le cas de

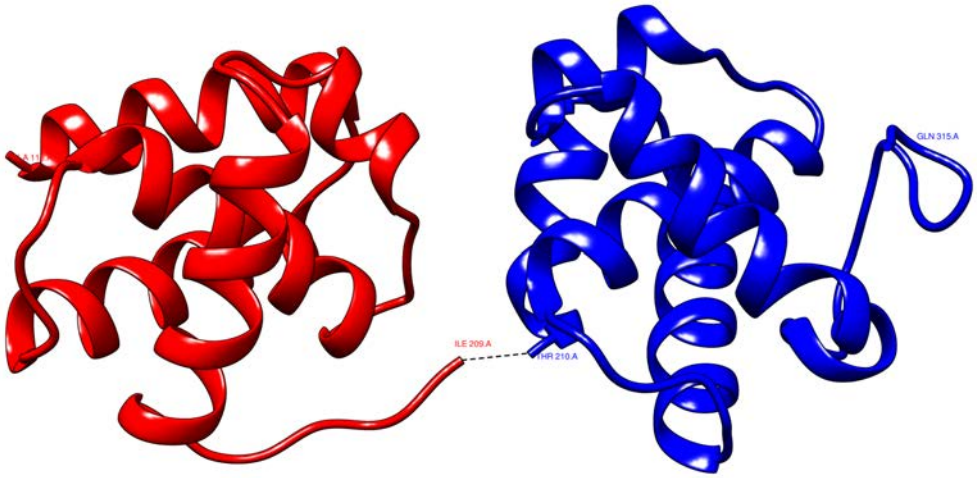



FIGURE 1.31 – Les deux domaines de la chaîne A de la protéine **1C9B** tels que découpé par les protocoles de CATH [90], [91]. La jonction entre les deux domaines se fait entre le résidu 209 (isoleucine, au centre en rouge) et le résidu 210 (thréonine au centre en bleu) et est symbolisée par les pointillés noirs.

**1SFT**, une alanine racemase [97] dont les domaines de la chaîne A sont insérés l'un dans l'autre (figure 1.32)

### 1.6.1 Brève analyse de la répartition des domaines structuraux discontinus au sein d'une base de données hiérarchique

Une partie de cette thèse est consacrée à la classification automatique des domaines structuraux au sein d'une base de données hiérarchique. Les méthodes présentées, qu'elles soient manuelles ou automatiques, approchées ou exactes, ont le défaut de nécessiter une grande quantité de ressources et de comparaisons. Nous avons donc observé la dispersion des domaines discontinus dans l'une de nos bases de données de référence afin de constater un regroupement, ou non, des domaines discontinus dans une section plutôt qu'une autre de la base de données. Si cette dispersion s'avère non aléatoire, nous pourrons l'utiliser dans le cadre de notre classification pour élaguer des parties entières de la base de données hiérarchisée. En effet, si le domaine que l'on cherche à assigner est continu, en cas de répartition non aléatoire il pourra être possible de retirer les ensembles de domaines discontinus.

Les figures suivantes sont des observations centrées sur l'une des bases de données recensant et classant les domaines structuraux : CATH [87]. Une présentation plus complète de CATH a été faite dans le chapitre 2. En résumé, la version 4.0.0 de CATH recense 235858 domaines structuraux répartis en 4 classes. Ces classes contiennent des sous-groupes, nommés *Architectures*, et les architectures des sous-groupes : *Topologies*. Nous avons regardé

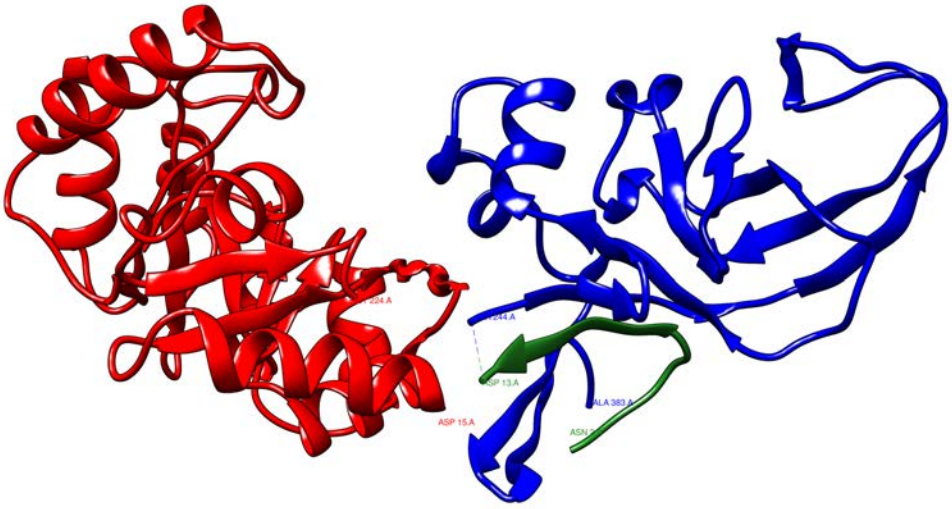



FIGURE 1.32 – Les deux domaines de la chaîne A de la protéine **1SFT** tels que découpés par les protocoles de CATH [87], le domaine 1sftA01 (en rouge, résidus ASP 15 à MET 224) est inséré dans le domaine 1sftA02 (en bleu et vert) qui est composé des premiers résidus (ASN 2 - APS 13, en vert) et de la fin de la chaîne polypeptidique (ALA 244 - ALA 383, en bleu).

la répartition des domaines structuraux au sein de ces différents groupes pour avoir une information quant à la fréquence d'apparition des domaines discontinus. Les domaines structuraux discontinus se retrouvent indifféremment dans les quatre classes définies par CATH (cf figure 1.33) selon un pourcentage allant de 2 à 20%.

La figure 1.34 illustre la variété des différents groupes de domaines structuraux. La présence ou non d'insertions de domaines ne semble pas liée à l'architecture considérée et une étude approfondie des topologies (les repliements ou folds) notamment dans les groupes contenant beaucoup de membres pourrait fournir des informations supplémentaires sur ces groupes au niveau purement structurel, mais également au regard de l'évolution.

Les domaines structuraux discontinus représentent 15% des domaines répertoriés dans CATH, l'étude de leur répartition au sein des différents groupes de la classification peut permettre de mieux comprendre les évènements évolutifs ayant pu intervenir sur les protéines.

Ces études préliminaires montrent que les domaines discontinus sont présents un peu partout. En continuant plus profondément les analyses il serait possible de trouver des sous-groupes où, de manière significative il y a présence ou absence de domaines discontinus, mais pas dans les hauts niveaux observés. En conclusion, à partir de ces seules informations, la continuité d'un domaine ne peut pas servir de critère dans le cadre de la classification de domaines structuraux.

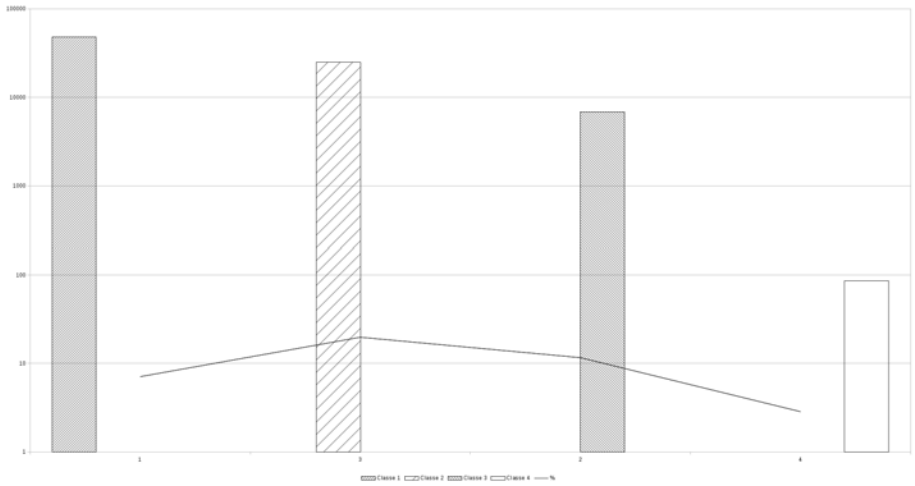



FIGURE 1.33 – Pourcentage de domaines structuraux discontinus au sein des classes de CATH et nombres de domaines recensés dans la version 4.0.0
Les domaines discontinus sont présents dans les quatre classes, par conséquent on ne peut utiliser le critère de continuité pour élaguer des ensembles de domaines à ce niveau.

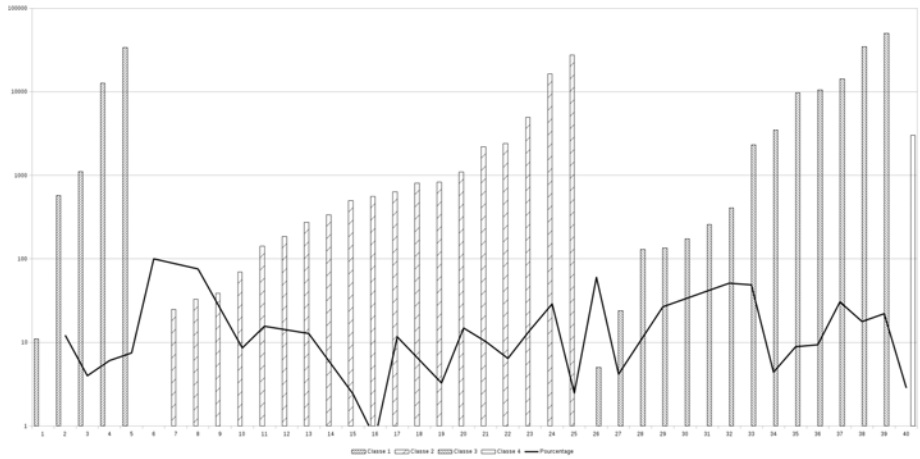



FIGURE 1.34 – Pourcentage de domaines structuraux discontinus au sein des architectures de CATH et nombres de domaines recensés dans la version 4.0.0
Les architectures sont triées par classe et nombre de membres.
Certaines architectures ne contiennent pas de domaines structuraux, l'une d'elles (première sur le graphique) ne peut pas être considérée comme significative en raison du faible nombre de membres. En revanche, l'architecture étiquetée 16 possède un nombre de membres assez grand pour que l'on puisse songer à un trait caractéristique. Néanmoins, cela demanderait une analyse plus poussée de l'architecture pour le valider.

### 1.6.2 Site catalytique d'un domaine protéique

Certaines protéines catalysent des réactions, elles sont nommées enzymes. La portion spécifique de l'enzyme qui effectue cette action se nomme le *site actif* ou *site catalytique*. Cette zone peut n'être composée que de quelques résidus : ainsi une mutation dans un site catalytique peut avoir de grosses répercussions sur la fonction de la protéine, allant de la modification de la fonction à son inhibition. La fonction d'une protéine est entre autres portée par la structure, mais la biochimie notamment au niveau catalytique est également cruciale.

### 1.6.3 Sites de liaisons

Les sites de liaisons sont des régions spécifiques en surface de la protéine pouvant former des liaisons chimiques avec d'autres protéines, ADN, ARN ou encore petites molécules (ligands), et donc permettre à la protéine d'interagir avec ces molécules [23]. Les sites de liaisons de deux protéines interagissantes sont appelés interfaces protéine-protéine (encore interface protéine-ligand dans le cas d'interactions protéine-molécule). La compréhension du fonctionnement de ces interfaces, de ces interactions entre protéines est un domaine de recherche ouvert [78]. L'augmentation du nombre de données structurales a permis l'expansion du nombre de méthodes de prédictions.

Il existe plusieurs approches de prédiction de sites de liaisons, l'une d'elles analyse des ensembles de sites connus (au niveau de complexes protéiques) pour les caractériser puis recherche ces sites à la surface d'une nouvelle protéine [62]. Certaines méthodes de prédiction sont basées sur l'observation que les résidus présents dans les sites de liaison sont plus conservés que le reste des résidus même si cette information de séquence n'est pas suffisante pour une prédiction complète d'une interface entre deux protéines [21].

## 1.7 Modularité et plasticité des protéines

### 1.7.1 Modularité des protéines multidomaines

Si les acides aminés sont les briques du vivant, les domaines structuraux sont les briques des protéines. Une protéine est constituée d'un ou de plusieurs domaines structuraux, ré arrangeables. Ce concept de modularité au sein des protéines est très important pour les évolutionnistes pour qui les domaines sont des séquences qui peuvent apparaître à différents endroits et plusieurs fois dans les génomes. Ainsi un domaine peut apparaître au sein de différentes protéines et une protéine peut être multidomaines. L'origine de cette modularité s'explique au niveau des génomes par les recombinaisons, les translocations, inversions, duplications et autres délétions de séquences [16]. Les erreurs lors de la duplication et les répétitions internes [15] sont aussi une explication de la modularité des protéines sans oublier les transferts horizontaux (entre espèces) de gènes.

Ainsi, observer ces domaines qui sont considérés comme des unités modulaires des protéines permet une étude phylogénétique et une compréhension au niveau évolutif du vivant.

### 1.7.2 Plasticité des protéines

La plasticité des génomes est leur capacité à supporter des modifications telles que les duplications, les translocations et autres évènements évolutifs. Ainsi les gènes codant pour les protéines évoluent plus ou moins au cours du temps, subissant des modifications ou réarrangements neutres ou altérants. Au niveau structural ces évènements sont également à l'origine de *permutations circulaires*, qui correspondent à une modification de l'ordre des résidus dans la séquence protéique d'une structure à l'autre, mais aussi de l'apparition de *charnières*, c'est-à-dire d'un coude présent dans une structure et non dans une structure homologue, ou encore à la constitution de protéines composées uniquement d'un motif structural répété.

Ces trois cas sont des défis pour les outils de comparaisons de structures (discuté dans les chapitres 2 7).

### 1.7.3 Permutations circulaires

**Définition 1.4 (Permutation circulaire)** *Une permutation circulaire est un changement d'ordre des résidus d'une protéine à l'autre. Ici, l'ordre est l'enchaînement linéaire des AA tel que défini par la séquence primaire. L'extrémité N-terminale de l'une correspond à l'extrémité C-terminale de l'autre.*

Le premier cas fut observé en 1979 [32] puis de nombreux autres cas sont apparus dans la littérature [72, 114]. Ces modifications dans la séquence n'entraînent que peu de modifications dans la structure. Ainsi les structures sont similaires alors que linéairement les séquences divergent.

Des bases de données comme BALIBase [13] ou encore SISYPHUS [8] recensent une partie des cas de permutations circulaires.

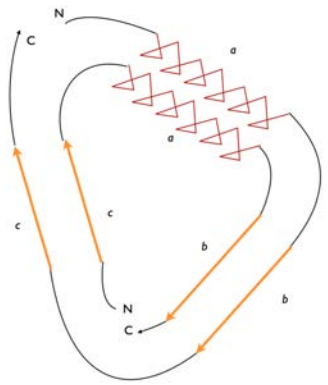



FIGURE 1.35 – Représentation schématique de permutation circulaire issue de [18]

Les permutations circulaires ne sont pas automatiquement détectées par les algorithmes de comparaisons de structures. En effet, les protéines étaient initialement supposées linéaires, les duplications, insertions et autres réarrangements n'étaient pas connus.

### 1.7.4 Charnières

Les protéines sont des objets flexibles, de la vibration atomique aux larges mouvements de domaines en passant par les rotations de chaînes latérales. Ces mouvements ont un rôle important dans la fonction de la protéine, notamment dans le cas des enzymes avec des changements de conformations selon la présence/absence du ligand. La conformation d'une protéine est la forme qu'elle adopte dans l'espace. Certaines portions de la protéine bougent autour d'un ou plusieurs points d'inflexion nommés *charnières* [37, 117].

**Définition 1.5 (Charnières)** *Les charnières sont des points de rotation desquels des portions de protéines se meuvent de manière plus ou moins large.*

La figure 1.36 présente deux structures identiques au niveau de la séquence comme le montre la matrice de points résultant de l'alignement des séquences avec BLASTp, mais dont la structure globale est très différente à cause d'une charnière. De fait, comme le montreront des analyses ultérieures, certains outils d'alignements ne réussissent pas à détecter la similarité entre ces deux structures.

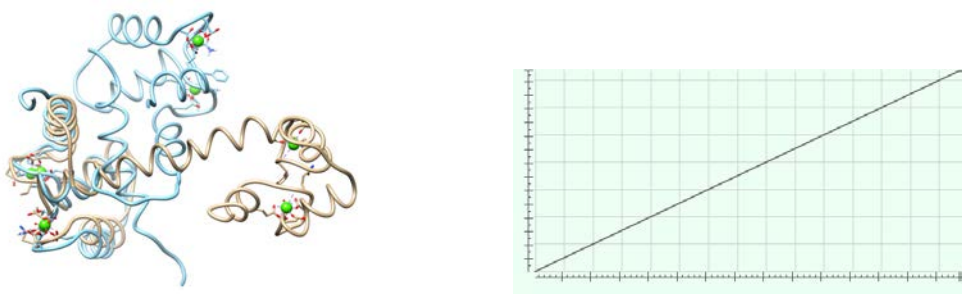


FIGURE 1.36 – Superposition de **4CLN,A** et **2BBM,A** et alignement issu de BLASTp (matrice de points)

Il existe des outils dédiés à la recherche de ces charnières comme HingeProt [35] ou FATCAT [119]. Nous nous sommes intéressés à ces charnières afin d'observer les capacités de nos outils à détecter ces cas de figure.

### 1.7.5 Répétitions structurales internes

La proportion de répétitions au sein des protéines est estimée à 25% [76]. Ces répétitions allant de quelques segments d'acides aminés à de larges domaines sont issues de duplications et de recombinaisons de portions de gènes [6]. Le nombre de ces répétitions et leur arrangement jouent un rôle important dans la fonction des protéines, ces protéines assurant des rôles de transport, d'assemblage de complexes protéine-protéine et de régulation. Si certaines protéines contiennent des répétitions d'autres en sont presque exclusivement constituées, c'est notamment le cas des protéines solénoïdes. Une protéine solénoïde est une protéine comprenant des motifs structuraux répétés arrangés de manière à former une super-hélice continue

[60]. Plusieurs types de domaines protéiques sont regroupés sous le terme solénoïde : domaines avec répétitions riches en leucine (LRR), les répétitions armadillo, les domaines à répétitions ankyrine (ANK) ou encore les domaines à répétition HEAT. Les solénoïdes étant des répétitions du même motif structural, on s'attend à une identité de séquence assez forte ce qui est vrai pour la plupart des cas. Cela s'explique par la conservation corrélée de la structure primaire (séquence) et de la structure tertiaire des protéines, ce même si les contraintes de conservation sont plus importantes au niveau de la structure tertiaire [103].

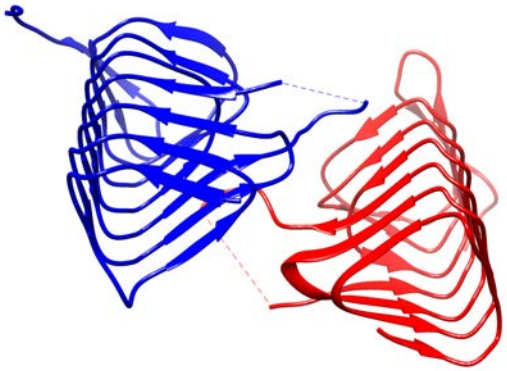


FIGURE 1.37 – Exemple de protéine solénoïde : **3TV0**

## 1.8 Une famille d'enzymes : les glycosides hydrolases, famille 5 (GH5)

Les glycosides hydrolases, famille 5 (ou plus simplement GH5), forment l'une des plus grandes familles de GH recensée sur CAZy [1]. Cette famille est associée à de nombreuses annotations fonctionnelles notamment la fonction endoglucanase, exoglucanase ou encore $\beta$-glucosidase. Les différentes activités enzymatiques connues des GH5 sont répertoriées dans le tableau 1.4 avec leurs numéros EC.

**Classification EC** La classification EC (Enzyme Commission) [2] recense les différentes fonctions pouvant être catalysées par les enzymes selon un système numérique à quatre nombres, le premier symbolisant la catégorie de réaction catalysée (1. Oxydo-réductases, 2. Transférases, 3. Hydrolases, 4. Lyases, 5. Isomérases et 6. Ligases), la seconde le substrat (général), le troisième le substrat spécifique et un numéro de série.

Les GH5 sont des hydrolases (EC 3.x.x.x), c'est à dire qu'elles catalysent la réaction consistant à rompre une liaison au sein d'une molécule $(R - R')$ via une molécule d'eau $(H_2O)$ et ainsi provoquer la formation des molécules $R$ et $R'$ ($R$ et $R'$ étant deux glucides,

1. http://www.cazy.org/GH5.html
2. http://www.chem.qmul.ac.uk/iubmb/enzyme/

TABLE 1.3 – Liste des EC référencées chez les GH5

| **Enzyme** | **Numéro EC** |
|---|---|
| endo-$\beta$-1,4-glucanase / cellulase | EC 3.2.1.4 |
| endo-$\beta$-1,4-xylanase | EC 3.2.1.8 |
| $\beta$-glucosidase | EC 3.2.1.21 |
| $\beta$-mannosidase | EC 3.2.1.25 |
| $\beta$-glucosylceramidase | EC 3.2.1.45 |
| glucan $\beta$-1,3-glucosidase | EC 3.2.1.58 |
| licheninase | EC 3.2.1.73 |
| Exo-$\beta$-1,4-glucanase / cellodextrinase | EC 3.2.1.74 |
| glucan endo-1,6-$\beta$-glucosidase | EC 3.2.1.75 |
| mannan endo-$\beta$-1,4-mannosidase | EC 3.2.1.78 |
| cellulose $\beta$-1,4-cellobiosidase | EC 3.2.1.91 |
| steryl $\beta$-glucosidase | EC 3.2.1.104 |
| endoglycoceramidase | EC 3.2.1.123 |
| chitosanase | EC 3.2.1.132 |
| $\beta$-primeverosidase | EC 3.2.1.149 |
| xyloglucan-specific endo-$\beta$-1,4-glucanase | EC 3.2.1.151 |
| endo-$\beta$-1,6-galactanase | EC 3.2.1.164 |
| hesperidin 6-O-$\alpha$-L-rhamnosyl-$\beta$-glucosidase | EC 3.2.1.168 |
| $\beta$-1,3-mannanase | EC 3.2.1.- |
| arabinoxylan-specific endo-$\beta$-1,4-xylanase | EC 3.2.1.- |
| mannan transglycosylase | EC 2.4.1.- |

ou bien un glucide et un fragment non glucidique). La formule générale d'une hydrolyse est la suivante :

$$R - R' + H_2O \rightleftharpoons R - OH + R' - H \tag{1.1}$$

Les GH5 ne peuvent pas hydrolyser tous les types de molécules, elles sont spécialisées dans les ruptures de liaisons glycosidiques (d'où leur nom, glyscosidases et leur EC spécifique, 3.2.x.x) et par conséquent dégradent des glucides complexes. Les glycosides hydrolases ont été séparées en familles [46, 47] sur la base de comparaisons de séquences. On dénombre plus d'une centaine de familles de GH. Puis, grâce à la détermination de structures, certaines familles de GH partageant un repliement (fold) commun ont été regroupées dans des clans. Ainsi, les GH5 sont une famille de Glycosides Hydrolases appartenant au clan GH-A ( $(\alpha/\beta)_8$[86], exemple de structure avec **1H11**, fig 1.38 ).

Vingt et une fonctions enzymatiques ont été recensées chez les GH5, agissant sur autant de substrats différents. A l'heure actuelle, 6219 séquences protéiques ont été affectées aux GH5, réparties dans les grands groupes phylogénétiques de la manière suivante :

– 55 issues des Archées

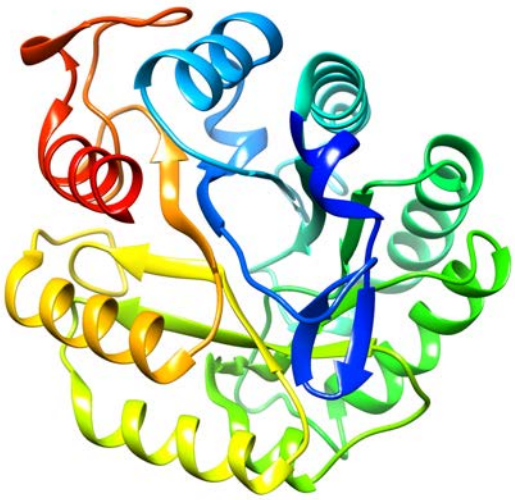


FIGURE 1.38 – Structure de **1H11**, une GH5 avec le repliement caractéristique $(\alpha/\beta)_8$
Le coeur de la structure est un ensemble de feuillets $\beta$ liés

- 4317 issues du règne bactérien
- 1731 séquences eucaryotes
- 2 séquences de virus
- 114 séquences non classifiées

En revanche, seules 57 structures d'enzymes sont disponibles, sous forme libre et/ou en complexe (ainsi plusieurs fichiers PDB sont disponibles pour une même enzyme) :

- 1 enzyme issue des Archées
- 40 issues du règne bactérien
- 16 entrées eucaryotes

### 1.8.1 Données test

Pour nos tests de recherche de divergences, nous avons utilisé un ensemble de 11 protéines issues de la famille des GH5, sous-famille 4.

## 1.9 Discussion, relation séquence, structure, fonction

Une séquence implique une structure et définit une fonction. Tel est le paradigme séquence, structure, fonction (SSF) qui a longtemps été communément admis par les scientifiques. De même, des séquences similaires impliquent des structures similaires et donc des fonctions similaires.

Le dogme d'Anfinsen postule que la structure d'une protéine globulaire est définie par sa seule séquence. Si en effet la séquence d'une protéine joue un rôle évident dans sa future

TABLE 1.4 – Jeu de données GH5, sous-famille 4
Descriptions et annotations issues de CAZy

| PDBid,chaîne | Nom | Numéros EC |
|---|---|---|
| 3NDY,A | Endoglucanase D | 3.2.1.4 |
| 4IM4,A | Endoglucanase E | 3.2.1.4 |
| 3AYS,A | Endoglucanase | 3.2.1.4 |
| 4W85,A | Xyloglucan-specific endo-beta-1,4-glucanase | 3.2.1.151 |
| 1EDG,A | Endoglucanase A | 3.2.1.4 |
| 4V2X,A | $\beta$-glucanase (CelB ;BhCel5B ;BH0603) (Cel5B) | 3.2.1.4 |
| 2JEP, A | Xyloglucanase (XG5 ;PpXG5) | 3.2.1.151 |
| 3ZMR,A | Endo-xyloglucanase (BoGH5A ;BACOVA_02653) | 3.2.1.151 |
| 3VDH, A | endo-$\beta$-1,4-glucanase (Orf4 ;CMCase) (Cel5A) | 3.2.1.4 |
| 4NF7,A | Butyrivibrio proteoclasticus B316 (bpr_I1710 (Cel5C)) | - |
| 3AYR,A | endoglucanase A (CelA ;EglA) (partiel) | 3.2.1.4 |

structure, et si, dans la majorité des cas, des protéines aux séquences similaires ont des structures similaires, deux protéines peuvent avoir des structures très similaires avec des séquences très différentes. C'est le cas par exemple des protéines **1EDE** et **1CQW** (voir figure 1.39) qui appartiennent toutes deux à la famille des alcanes déshydrogénases (EC 3.8.1.5). Elles sont structurellement très proches, mais séquentiellement très éloignées (moins de 30% d'identité).

Cette convergence structurelle a plusieurs explications : il faut tout d'abord savoir que la taille de l'espace structural est finie et bien plus petite que celle de l'espace des séquences. C'est-à-dire que le nombre de repliements possibles est plus faible que le nombre de séquences potentiellement possibles à générer. Dans [66], E. Krissinel a conclu que la structure est tolérante aux substitutions de résidus si celles-ci conservent les propriétés physico-chimiques. Krissinel observe également que certaines mutations sont plus sujettes à déformer une structure, celles ciblant des résidus impliqués dans des liaisons stabilisantes (ponts salins, ponts hydrogènes). Cette stabilité de la structure malgré la flexibilité de la séquence permet un maintien de la fonction au cours de l'évolution.

Et de même que précédemment, l'hypothèse de deux structures similaires menant à des fonctions similaires est largement admise. Admise et généralement vérifié, mais, il existe des contre-exemples comme (**1UMZ,A** vs **2UWC,A**) qui ont une séquence et une structure très similaires, mais l'une est une hydrolase et l'autre une ligase, deux fonctions antagonistes. Toujours dans la catégorie contre-exemples, artificiels cette fois, il existe des protéines avec des séquences similaires (qui devraient logiquement impliquer des structures similaires) qui se replient différemment dans l'espace.

Enfin, que dire des protéines intrinsèquement désordonnées ? Ces protéines ne se replient nativement que partiellement, voire pas du tout, et pourtant ont différentes fonctions et non des moindres (elles agissent notamment au niveau de la reconnaissance moléculaire). De fait, elles se cristallisent peu ou pas, un cauchemar pour les biochimistes et les biologistes

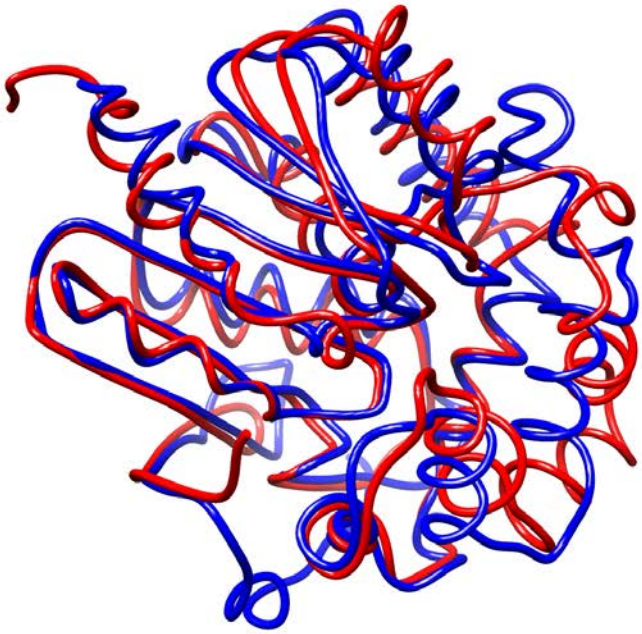


FIGURE 1.39 – Superposition des squelettes des protéines **1EDE** et **1CQW** avec l'outil TM-Align, TM-score égal à 0.80520 ($0 \leq TM - Score \leq 1$) pour un pourcentage d'identité de séquence des résidus alignés égal à 0.273)

structuraux. La découverte de ces protéines a pratiquement fait exploser le paradigme SSF tant par leur flexibilité que par leur grande représentation au sein des génomes.

En conclusion, les relations entre la séquence d'une protéine, sa structure et sa fonction ne sont pas actuellement clairement connues. Les cas simples ont été étudiés et permettent, par comparaison, d'augmenter la connaissance des protéines nouvellement découvertes, mais il reste tout un champ de recherche dans le domaine de l'aide à la compréhension des cas non évidents.

La grande flexibilité des protéines, leur modularité en fait des objets complexes à étudier.

## 1.10 Conclusion.

Ce chapitre d'introduction aux protéines a permis de définir les notions de base comme la *structure* d'une protéine, le cœur de cette thèse. Étudier la structure d'une protéine permet de récolter de précieuses informations concernant la compréhension du vivant. Les protéines sont composées de briques : les acides aminés, et d'unités modulaires compactes et stables : le domaine. Ces domaines se retrouvent à travers les espèces dans différentes protéines. Les évènements évolutifs (duplications, translocations, indels ...) entraînent des modifications

chez les protéines telles que de grandes flexibilités ou encore des inversions de séquences ne modifiant pas la structure (dans le cas des permutations circulaires). Les domaines structuraux tels que définis ici servent de base aux classifications et protocoles d'assignations de protéines à un groupe (chapitres 2, 3) Ces chapitres présentent l'état de l'art, les outils et les bases de données qui classifient ces domaines structuraux ainsi que nos méthodes et nos protocoles de classification. L'étude des modifications induites par les changements au niveau des génomes (permutations circulaires, charnières, répétitions) seront l'objet des chapitres 4 à 8 Après un chapitre d'état de l'art consacré notamment à la présentation des outils servant à analyser la qualité de nos méthodes nous présentons ces méthodes. Elles sont au nombre de trois, les deux premières étant deux modules fonctionnant de concert pour détecter des permutations circulaires ou des charnières. Le troisième outil est lui dédié aux répétitions structurales. Puis nous traitons les différents cas de changements au travers d'exemples et d'études via des benchmarks. Nous avons également mis en avant les groupes fonctionnels des protéines ($FGS$), les cinq types présentés se retrouvent dans les acides aminés ce qui autorise des substitutions d'acides aminés dans la séquence sans modifier la présence d'un groupe fonctionnel dans la structure. L'étude de ces groupes fonctionnels est le thème du chapitre 9, cela via des protéines issues de la famille des GH5.

# Deuxième partie

# Classification structurale de protéines, comparaison globale de structures

# Chapitre 2

# État de l'art

## 2.1 Introduction

Découvrir la fonction des protéines est un vaste sujet d'étude depuis l'identification des premières protéines. Il est possible de la déterminer dans certains cas par expérimentation ou d'autres cas, par prédiction en se basant sur la comparaison de la protéine à fonction inconnue avec d'autres protéines de fonction connue. Le regroupement des protéines aux fonctions similaires et la caractérisation de ces groupes ont motivé l'émergence des classifications. Ces classifications étaient initialement basées sur les comparaisons de séquences : des séquences proches indiquant une relation d'homologie, certaines bases de données comme PROSITE[54] combinent les protéines de la base de données de séquences UniProt [11, 10] et en extraient de nombreuses informations fonctionnelles et en termes de regroupement de séquences. L'une des bases de l'étude des séquences protéiques est la relation d'homologie qui unit deux séquences très proches. Et deux séquences proches sont logiquement amenées à avoir des structures proches et donc des fonctions proches. C'est cette logique qui est derrière les classifications, regrouper ce qui se ressemble pour déduire des informations fonctionnelles. Au niveau structurel, la similarité de structures associée à une faible similarité de séquence était interprétée comme une relation d'analogie mais des études plus récentes ont montré que la similarité de structures tendait en certains cas à refléter la présence d'un ancêtre commun lointain.

Si historiquement parlant, les classifications sont basées sur les séquences protéiques connues, cela est de par le nombre beaucoup plus important de séquences que de structures. L'augmentation du nombre de structures protéiques dans la PDB [14] (autour de 90 000 structures sont disponibles en 2014) a permis des études à plus grande échelle et l'établissement de classifications de structures. Plus spécifiquement, les classifications structurales de protéines sont des classifications de domaines structuraux (une protéine pouvant être composée de plusieurs domaines et donc apparaître plusieurs fois dans la classification), qui consistent à regrouper ces domaines selon des critères purement structuraux puis des critères d'homologie afin de faire émerger des similitudes et ainsi aider à la compréhension de l'univers des protéines.

La classification des structures permet d'organiser les domaines structuraux et d'extraire

des différents groupes des caractéristiques propres et d'évaluer entre autres la plasticité du groupe. Les principaux objectifs des classifications sont (i) proposer une vue de l'évolution à travers les différentes hiérarchies menant aux familles protéiques et représenter les relations évolutives entre les protéines ; (ii) comprendre le rôle fonctionnel des protéines.

Nous nous sommes focalisés sur le problème de classification des structures protéiques.

Les bases de données hiérarchisées comme SCOP [82] et CATH [88] ont pour objectif de caractériser l'univers des protéines, de l'organiser, avec des protocoles manuels pour SCOP et semi-automatiques pour CATH [89]. Par conséquent, l'intégration de nouvelles structures est un processus long et coûteux. C'est pour cela que la majorité des structures de la PDB restent non classée à l'heure actuelle.

Le but de la première partie de cette thèse, dédiée à la classification, n'est pas de créer une nouvelle classification mais d'enrichir les classifications existantes de manière fiable en résolvant un problème d'identification des superfamilles (SFIP ou *Super Family Identification Problem*). De même le problème peut être considéré à un niveau structurel plus bas, on parle alors du FIP ou *Family Identification Problem*. Le niveau de superfamille correspond au niveau des classifications discriminant le repliement des domaines. Pour une classification donnée, il s'agit de classer un nouveau domaine (dit requête) au bon endroit dans la classification (soit dans la bonne superfamille).

L'une des approches utilisées est la recherche du plus proche voisin basée sur le principe qu'une protéine appartient à la même famille que la protéine dont elle est la plus proche structurellement. La méthode classique (méthode *one-to-all comparisons*) requiert de choisir un outil de comparaison de structures, son score associé, et pour un domaine requête donné, de comparer cette requête avec l'ensemble des domaines de la classification existante. Cette méthode contient trois principaux défauts : le nombre de comparaisons à effectuer (augmentant avec la taille des bases de données), la pertinence de l'outil de comparaison et le problème de caractérisation de l'espace des protéines. Nous reviendrons plus en détails sur ces défauts dans ces chapitres. Au vu de ces problèmes, nous avons élaboré des méthodes alternatives de classification avec pour objectif de : (i) supprimer la contrainte de *one-to-all*, (ii) garantir l'exactitude du résultat , (iii) s'affranchir des alignements structuraux pour se focaliser sur un score de comparaison, (iv) évaluer nos méthodes.

Nous commencerons par un aperçu de la variété des outils et scores de comparaison, et une présentation de quelques outils et scores de comparaison de structures qui vont nous servir soit de comparateurs pour nos méthodes soit directement au sein de nos méthodes. Puis nous décrirons la méthode *one-to-all comparisons* ainsi que les nôtres et leurs innovations. Ces innovations ont permis de répondre positivement à nos objectifs. Nous avons utilisés quelques jeux de données de tests et les résultats ont montré les points forts de nos méthodes dont une caractérisation de l'espace des protéines par une mesure métrique mais également quelques limites.

## 2.2 Classification hiérarchique des domaines structuraux

### 2.2.1 SCOP, Structural Classification Of Proteins

La classification SCOP est manuelle, basée sur l'inspection visuelle des domaines structuraux (avec l'aide d'outils de comparaison) ainsi que sur les informations relatives aux domaines issus de la littérature, [83, 73, 19].

SCOP contient quatre principaux niveaux de classification :

- Class, les domaines sont regroupés selon leur composition en structures secondaires.
- Fold (repliement), deux domaines ont le même repliement s'ils appartiennent à la même classe et si l'arrangement spatial et la connectivité de leurs structures secondaires est la même.
- Superfamily, une superfamille contient des membres structurellement proches et partageant un ancêtre commun.
- Family, les membres d'une même famille ont soit des structures et des fonctions très proches, soit une similarité de séquence supérieure à 30%.

La version 1.75 de SCOP (2009) contient 38221 protéines et 110800 domaines structuraux. Le prototype d'une nouvelle base de données, SCOP2 [7], est disponible depuis 2013. La structure hiérarchique n'est plus linéaire, un groupe pouvant appartenir à plusieurs groupes parents. Cela est dû à l'ajout d'informations comme les évènements évolutifs ou le type de protéines (qui n'apparaissent pas dans SCOP).

### 2.2.2 CATH

Développée par le groupe Orengo [1], elle est composée de quatre niveaux principaux dans lesquels elle classe les domaines protéiques [96, 90] :

- Class, les domaines sont séparés en quatre classes suivant leur composition en structures secondaires (hélices $\alpha$ et feuillet $\beta$).
- Architecture, à ce niveau, les groupes se caractérisent par leur arrangement spatial global, c'est-à-dire l'orientation des structures secondaires des domaines.
- Topology ou Fold family, classe les domaines selon leurs repliements en tenant compte de la connectivité des structures secondaires.
- Homologous superfamily, à ce dernier niveau, les domaines d'un même groupe sont estimés issus d'un même ancêtre commun. L'estimation est faite de manière manuelle, l'expert vérifie la présence de preuves d'une relation évolutive entre les membres du groupe.

Dans les niveaux suivants, SOLID, la classification n'est plus basée sur la similarité de structure mais sur la similarité de séquence des domaines (S : $similarité \geq 35\%$, O : $similarité \geq 60\%$, L : $similarité \geq 95\%$, I : $similarité = 100\%$, D : domaines uniques).

L'identification de nouveaux domaines à partir de protéines et leur classification au sein de CATH sont schématisées dans la figure 2.1, extraite de l'article de Greene *et al.* [41].

1. http ://www.cathdb.info/wiki/doku/ ?id=cathteam :index

Ce protocole contient une série d'étapes automatiques mais également deux étapes manuelles lorsque le processus automatique n'obtient pas de résultats assez fiables. Grâce à ce protocole, la version 4 de CATH (mars 2013) contient 235 858 domaines structuraux répartis en 2738 superfamilles (niveau H) pour 69 058 protéines annotées (la PDB en contient 100 450 en avril 2015).

### 2.2.3 Enrichir les bases de données hiérarchiques, problème d'identification des familles protéiques

La problématique est ici l'insertion d'un nouveau domaine structural au sein d'une classification hiérarchique. SCOP et CATH sont les deux principales classifications hiérarchiques de domaines structuraux (il existe également FSSP Families of Structurally Similar Proteins qui a classé les protéines en groupes selon les résultats paire à paire de l'algorithme DALI). L'ensemble des domaines structuraux classés de la même manière est regroupé dans un jeu de données nommé SCOPCATH [31]. L'inconvénient majeur de ces classifications est la partie manuelle de l'insertion de nouveaux domaines car cela est coûteux en temps et nécessite des experts compétents. Cette insertion de domaines dans la hiérarchie est délicate principalement dans les niveaux superfamilles et se retrouve sous le nom de Family Identification Problem (**FIP** ), le **problème d'identification des familles** et, un niveau au-dessus, le **problème d'identification des superfamilles** (**SFIP** ). Ce problème est plus difficile que le précédent puisque les domaines structuraux d'une même superfamille peuvent avoir très peu de similarités au niveau de leur séquence.

Par conséquent, l'un des grands challenges en biologie structurale est le développement de méthodes et algorithmes rapides, efficaces et automatiques d'assignation d'un domaine structural à une famille.

SCOP et CATH sont largement admises et utilisées par la communauté scientifique malgré leurs divergences mais le **problème de classification des domaines structuraux** reste ouvert. La principale difficulté dans la création d'une classification est la détection des similarités structurales entre domaines et l'identification de relations de parenté. Cela nécessite de trouver une mesure de similarité permettant de caractériser l'espace des domaines structuraux. Dernier souci et non des moindres, il est nécessaire de comparer chaque domaine à tous les autres domaines du jeu de données pour déterminer une classification fiable. Le problème d'intégration de nouveaux domaines structuraux au sein des classifications existantes est ici nommé **problème d'identification des familles protéiques** (définition 2.1). Ce problème peut être défini comme suit :

**Définition 2.1** *Problème d'identification des familles protéiques (FIP)*
*Soit un ensemble de domaines structuraux requêtes $Q = \{q_1, q_2, \ldots, q_n\}$, un ensemble de domaines structuraux cibles classés $T = \{t_1, t_2, \ldots, t_m\}$ et une fonction mesurant la similarité de deux domaines $S : Q \times T \rightarrow R^+$, le problème d'identification des familles protéiques consiste à assigner à chaque requête $q_i \in Q$ la classe de la cible dont elle est la plus proche voisine $NN_i$, voisine définie par l'équation 2.1*

$$NN_i = \underset{t_j \in T}{argmax} S(q_i, tj) \tag{2.1}$$

Ainsi, insérer un nouveau domaine (requête) au sein d'une classification existante revient ici à rechercher le domaine (cible) au sein de la classification dont la similarité avec la requête est maximale. Cette recherche nécessite donc la comparaison de la requête à toutes les cibles de la base. Cela induit très rapidement un nombre de comparaisons (notées **instances**) considérable. Deux stratégies de calculs de scores de similarité existent : méthode heuristique ou exacte. Les heuristiques produisent des résultats rapidement mais leur optimalité n'est pas prouvée et les méthodes exactes peuvent ne pas réussir à trouver une solution optimale en un temps raisonnable.

## 2.3 Estimer la similarité structurale entre deux protéines

L'estimation de la similarité entre deux protéines trouve sa source dans la recherche de la quantification de la ressemblance d'une structure par rapport à l'autre, tant au niveau parenté (homologie, origine commune), qu'au niveau fonctionnel. Deux structures proches supposent ainsi une fonction proche de par le paradigme structure-fonction. De nombreuses méthodes tentent de capturer cette ressemblance en mesurant la similarité ou la distance entre deux structures. Il en existe trois grands types :

- les mesures de distances entre matrices de distances inter-résidus,
- les mesures de recouvrement de cartes de contacts,
- la mesure de la déviation globale (RMSDc 2.4) des deux protéines après superposition optimale des deux structures,

Toutes ces méthodes retournent des **scores** censés illustrer la ressemblance entre deux protéines. Soit une protéine $P_1$ comparée à deux protéines $P_2, P_3$ via la méthode

$$S : P_1 \times P_2 \rightarrow R^+$$

qui cherche à trouver la ressemblance maximale entre deux structures. Si le score de similarité entre $s(P_1, P_2)$ de $P_1$ et $P_2$ est plus grand que le score $s(P_1, P_3)$ de $P_1$ et $P_3$, alors on peut dire que la protéine $P_1$ ressemble plus à la protéine $P_2$ qu'à la protéine $P_3$.

L'un des points essentiels de ce chapitre est la notion de **score** . Le score correspond à une valeur chiffrée de la similarité de deux protéines. Toute la difficulté des méthodes de comparaison tient dans le regroupement des caractéristiques communes aux protéines dans cette seule valeur. C'est pourquoi de nombreux outils retournent en plus du score un alignement associé permettant de visualiser la similarité. Néanmoins, pour les problèmes dont il est question dans cette partie du manuscrit (classification et assignation des domaines structuraux au sein de classifications hiérarchiques), l'alignement n'est pas utilisé directement, seule sa longueur est considérée. L'étude des alignements est abordée dans la seconde partie. La comparaison globale de protéines et l'assignation de domaines structuraux au sein des classifications hiérarchiques nécessite de trouver des scores qui estiment correctement la similarité entre structures et soient contraignants, c'est-à-dire qu'à la lecture seule de ce score il soit possible de conclure quant à la ressemblance des structures étudiées. C'est la raison pour laquelle de nombreux scores sont normalisés ou au moins relatifs à la longueur des protéines étudiées.

Remarque : dans les sections suivantes, il est précisé pour chaque score le cas dans lequel il est meilleur, c'est-à-dire s'il tend à être maximisé ou minimisé.

### 2.3.1 Difficulté de la comparaison de deux structures

Tel qu'il est présenté ci-dessus, il suffit d'obtenir un score pour estimer la similarité de deux structures, mais de par nature, le problème de la comparaison de structures est complexe car une grande partie des versions du problème a été démontrée NP-difficile [40]. Cela implique que les algorithmes de résolution du problème sont soit conçus pour une version simplifiée du problème, renvoyant un résultat approché, ne pouvant être garanti comme optimal et de plus, comme Godzik [38] le fait remarquer, il n'y a pas qu'une seule solution au problème. Soit ces algorithmes sont exacts et explorent un problème combinatoire donc peuvent nécessiter des temps non raisonnables pour retourner une solution.

Les scores sont en majorité basés sur la recherche d'un alignement de deux structures qui va maximiser ce score. Le choix de l'alignement peut influer, plus ou moins fortement, le score qui lui est associé."

### 2.3.2 Scores basés sur les mesures de distances inter-résidus

Ces scores ont une certaine popularité car ils ne requièrent aucun calcul de superposition de structures, basés sur les distances entre résidus issus du même domaine. Les scores inter-résidus ont plusieurs critères, la nature des résidus considérés, la présence de gaps dans l'alignement produit, la mesure de la différence de distances. I. Wohlers a dédié une partie de sa thèse à l'étude et la comparaison de ces scores et en a conclu qu'ils étaient tous pertinents au regard de leur fonction objectif, cela les rend difficiles à comparer. Ces méthodes opèrent en premier lieu un passage de la 3D à la 2D, les matrices de distances étant une représentation en deux dimensions de la structure des protéines. Les matrices sont indépendantes de l'orientation des structures. Les structures similaires ont des distances inter-résidus internes similaires, les scores ici servent donc à mesurer la proportion de distances similaires. Parmi les scores issus de cette catégorie on peut noter le score de DALI [49], le RMSDd (qui, comme son homologue basé sur les coordonnées nécessite d'être pondéré par la longueur de l'alignement correspondant), ou encore le score de DAST.

**RMSDd** Le RMSDd (Root Mean Square Deviation based on *d*istances) est une mesure de déviation globale basée sur les distances entre résidus d'une même protéine comparées aux distances associées dans l'autre structure. Le RMSDd se calcule sur la base de toutes les distances entre paires de résidus issus d'un alignement. Soit $Ali = (P_{1,i_1}, P_{2,j_1}; P_{1,i_2}, P_{2,j_2}; \ldots; P_{1,i_m}, P_{2,j_m})$ un alignement quelconque de longueur $m$ des protéines $P_1$ et $P_2$ (de longueurs respectives $|M|$ et $|N|$). Soit $i, i', j, j'$ quatre indices tels que $i < i', j < j'$ et $i$(resp. $i'$) est aligné avec $j$(resp. $j'$). Le RMSDd se calcule la manière suivante :

$$RMSDd = \sqrt{\frac{1}{p} \sum_{i<i' j<j'} (d(P_{1,i}, P_{1,i'}) - d(P_{2,j}, P_{2,j'}))^2} \tag{2.2}$$

avec $p = \binom{m}{2}$ (le nombre de paires de distances alignées) et $d(P_{1,i}, P_{1,i'})$ (resp. $d(P_{2,j}, P_{2,j'})$) la distance entre les résidus i,i' (resp. j,j') de $P_1$ (resp. $P_2$.

**DALI** Le principe de DALI est de représenter deux structures protéiques par leurs matrices de distances inter-résidus centrés en leurs $C\alpha$ ou $C\beta$. Holm et Sander décrivent leur méthode comme le coulissement d'une matrice sur l'autre, les sous-structures similaires apparaissant comme des patchs, des zones aux valeurs proches deux à deux, d'une matrice à l'autre. Un peu comme des cartes au trésor qu'il faut superposer à la lueur d'une bougie pour en extraire le chemin complet, le coulissement d'une matrice sur l'autre détermine l'alignement optimal, le meilleur appariement de résidus. L'algorithme de DALI a deux étapes : la première est une comparaison de toutes les sous-matrices de taille 6 (matrices contenant les distances entre les résidus $(i, .., i+5) \in P_1$, $(j, ..., j+5) \in P_2$ de deux protéines $P_1, P_2$), appelées motifs de contacts. Les sous-matrices similaires sont stockées et servent de graines à la seconde étape qui étend ces motifs pour maximiser le nombre de paires de résidus alignés. Pour chaque alignement DALI calcule le DALI-score, un score optimisé et retourne l'alignement avec le plus grand score. DALI est une heuristique, il en existe une version exacte, DALIX [116]

**DAST** DAST, Distance-Based Aligment Search Tool, a été initialement développé par Noël Malo-Dognin au sein de l'équipe Symbiose [2]. Il est composé de deux étapes majeures : la modélisation des appariements possibles entre résidus de deux protéines dans un graphe puis recherche de la clique maximum croissante au sein de ce graphe. A partir des matrices de distances des protéines $P_1, P_2$, DAST calcule un score $s(d(P_{1i,i'}), d(P_{2j,j'})$ basé sur les distances dans $P_1$ et $P_2$. Cela pour chaque appariement de paires de résidus $ii', jj'$, $i, i' \in P_1, j, j' \in P_2$ en se basant sur une valeur seuil $\lambda$ (paramétrable) tel que :

$$s(P_{1i,i'}, P_{2j,j'}) = \begin{cases} 1 & si\ ||d(P_{1i,i'}) - d(P_{2j,j'})|| \leq \lambda \\ 0 & sinon \end{cases} \tag{2.3}$$

Soit G=(V,E) un graphe modélisant la comparaison de deux protéines $P_1, P_2$, $V$ contient l'ensemble des appariements de résidus $i, j$ issus respectivement de $P_1, P_2$ possibles et une arête $e_{v,w} \in E$ existe entre deux sommets, $v_{i,j}, w_{i',j'} \in V$ si et seulement si le score $s(P_{1i,i'}, P_{2j,j'}$ est égal à 1. Ensuite DAST recherche dans G une clique maximale croissante (i.e, pas de croisement des séquences au sein de l'alignement final : si le quatrième résidu de $P_1$ est aligné avec le sixième résidu de $P_2$ alors le cinquième résidu de $P_1$ ne peut être alignés avec le cinquième résidu de $P_2$) et renvoie l'alignement correspondant ainsi qu'un RMSDd associé.

### 2.3.3 Scores de similarités basés sur les structures superposées

De nombreuses méthodes comme TMalign [126], LGA [123], MICAN [80], mesurent les distances après superposition optimale des structures et retournent le plus grand alignement avec un score optimisé selon leur fonction objectif. Ce score prend bien souvent en compte

2. IRISA-INRIA Rennes Bretagne Atlantique

le RMSDc, la mesure de déviation globale entre les structures, qui est une valeur phare dans la comparaison de structures.

**Le coordinate Root Mean Square Deviation, ou RMSDc,** est sans conteste la mesure de qualité d'un alignement structural la plus utilisée, il est défini comme suit (équation 2.4 [43]) :

$$RMSDc = \sqrt{\frac{1}{N_e}\sum_{i=1}^{N_e}(d_{i,j'})^2} \tag{2.4}$$

avec $N_eG = (V, E)eq1$ le nombre de paires de résidus alignés, $d_{i,j'}$ la distance des résidus $i \in P_1$ et $j' \in P_2$ après superposition de la protéine $P_2$ sur la protéine $P_1$. Plus le RMSDc est faible, meilleur est l'alignement. Néanmoins il faut relativiser cette dernière affirmation car à RMSDc égal, la qualité sous-jacente est grandement différente entre un alignement de longueur 10 et un alignement de longueur 100. Le premier peut être considéré comme relativement mauvais et le second plutôt bon. Les deux valeurs (RMSDc et longueur) jouent dans l'interprétation du résultat.

**Le RMSD100 (équation 2.5)** , moins utilisé, tend à pallier le défaut du RMSDc via une normalisation grâce à la longueur de l'alignement ($N_e$).

$$RMSD100 = \frac{RMSDc}{1 + ln\left(\sqrt{\frac{N_e}{100}}\right)} \tag{2.5}$$

Tout comme le RMSDc, plus le RMSD100 est faible, meilleur est l'alignement.

**Le S score** combine longueur de l'alignement et RMSDc mais ne tient pas compte de la longueur des protéines comparées [61], il est utilisé notamment par SARF2 [3].

$$Sscore = \frac{3N_e}{1 + RMSDc} \tag{2.6}$$

**Le Q score (eq 2.7)** est employé au sein de l'algorithme de SSM (Secondary Structure Matching, [65])

$$Qscore = \frac{N_e^2}{|P_1||P_2|\left(1 + \left(\frac{RMSDc}{3}\right)^2\right)} \tag{2.7}$$

**Le Z score [52]**

$$Zscore = 0.25N_e * exp(-0.39(RMSDc)^2) \tag{2.8}$$

**Le TMscore [126] [118] (eq 2.9)** est basé sur la distance après superposition des paires de résidus alignés. Il est implémenté au sein de TMalign [126] notamment mais également calculé par d'autres outils comme MICAN [80]

$$TMscore = \frac{1}{N_e}\sum_{i=1}^{N_e}\frac{1}{1+\left(\frac{d_{i,j'}}{d_0(N_e)}\right)^2} \tag{2.9}$$

avec $d_0(N_e) = 1.24(N_e - 15)^{\frac{1}{3}} - 1.8$. Le TMscore varie entre 0 et 1. Lorsqu'il est supérieur à 0.5, les protéines partagent le même repliement [118].

**SAS est défini comme suit :**

$$SAS = \frac{RMSDc \times 100}{N_e} \tag{2.10}$$

avec Ne le nombre de résidus équivalents, score à minimiser [61].

**GSAS est défini par :**

$$SAS = \frac{RMSDc \times 100}{N_e - N_{gaps}} \quad siN_e > N_{gaps} \tag{2.11}$$

avec $N_{gaps}$ le nombre de gaps (sauts dans la séquence) dans l'alignement, score à maximiser [61]

**Le score MI (Match Index, eq 2.12 [61])** est un score géométrique ($MI \in [0,1]$)également basé sur le RMSDc et qui est meilleur lorsqu'il tend vers 1 (maximisation). Il est également normalisé ce qui permet de comparer deux alignements.

$$MI = 1 - \frac{1+N_e}{\left(1+\frac{RMSDc}{1.5}\right)(1+min(|P_1|,|P_2|))} \tag{2.12}$$

avec $|P_1|$ et $|P_2|$ le nombre de résidus de chaque protéine.

**L'index de similarité (SI, eq 2.13 [61])** est une mesure géométrique normalisée de la similarité de deux protéines. A l'inverse du MI, plus le SI est faible, meilleur est l'alignement.

$$SI = \frac{RMSDc * min(|P_1|,|P_2|)}{N_e} \tag{2.13}$$

### 2.3.4 Scores de similarités basés sur la longueur d'un alignement de séquences

La majorité des outils de comparaison de structures protéiques retournent un alignement. A partir de celui-ci, on peut déduire une série de scores de similarité comparables d'un outil à l'autre. Soient $P_1$ et $P_2$ deux protéines, $|P_1|$ (resp. $|P_2|$) est la longueur de la protéine $P_1$ (resp. $P_2$) et $N_e$ la longueur de l'alignement de $P_2$ avec $P_1$ issu d'un outil de comparaison. A partir de ces valeurs, nous pouvons établir trois scores de similarité $\sim$, présentés ci-dessous :

1. Proportion de résidus alignés sur le nombre moyen de résidus :

$$s_{sum} = \frac{2Ne}{|P_1| + |P_2|} \tag{2.14}$$

2. Proportion de résidus alignés sur le nombre minimal de résidus :

$$s_{min} = \frac{N_e}{min(|P_1|, |P_2|)} \tag{2.15}$$

3. Proportion de résidus alignés sur le nombre maximal de résidus :

$$s_{max} = \frac{N_e}{max(|P_1|, |P_2|)} \tag{2.16}$$

Ces scores pourraient servir de bases aux protocoles de classification présentés par la suite, néanmoins ils présentent plusieurs faiblesses : la première est l'absence de contrôle de la qualité de l'alignement fourni. La qualité du score va être totalement dépendante de l'outil de comparaison utilisé. C'est pourquoi un score basé sur un alignement local soumis à un seuil de RMSDc n'aura pas la même signification qu'un score basé sur un alignement global distance inter-résidus dépendant.

Ainsi, lorsqu'il est dit plus haut que les scores allaient être comparables d'un outil à l'autre, cela ne va être possible que dans le cadre d'une recherche globale telle une classification. C'est à dire, pour un même score de similarité, les deux outils vont-ils permettre d'aboutir à la même classification ? Une comparaison plus directe de deux outils n'est pas envisageable. Une autre faiblesse réside dans le choix du score de similarité, chacun reflète un aspect de la similarité des protéines considérées et est donc un candidat à l'intégration dans le protocole de classification.

**Similarité normalisée [61]** Cette similarité est définie comme suit :

$$Snorm = 100 \times \frac{2N_e}{|P_1| + |P_2|} \tag{2.17}$$

### 2.3.5 Recouvrement de cartes de contacts et mesures de similarité

Les protéines sont modélisées par une carte en 2 dimensions qui modélise des contacts entre résidus. Le principe de la méthode est d'associer les résidus de deux protéines de telle sorte que le nombre de contacts communs soit maximisé. Cela reflète une similarité géométrique entre les deux protéines considérées. Nous présenterons cette approche en détail ainsi qu'un outil (Apurva) implémentant cette approche et ses fonctionnalités qui permettent d'optimiser la mesure du nombre de contacts communs et différents scores de similarité qui en découlent.

Le recouvrement de cartes de contacts de protéines CMO, Contact Map Overlap, et le problème qui lui est associé : $maxCMO$ (la maximisation du recouvrement de cartes de contacts) ont été introduits par Godzik en 1996 [38].

Une carte de contacts est une représentation en deux dimensions d'une protéine soit sous forme de graphe (définition 2.2) soit sous forme de matrice binaire carrée $N^2$ où $N$ est le nombre de résidus constituants la protéine. Une case $i, j$ de la matrice correspond au contact entre les $i^{ème}$ et $j^{ème}$ résidus de la protéine. Si la distance entre ceux-ci est inférieure ou égale à un seuil $\mu$, alors la case contient la valeur 1 ; 0 sinon. La figure 2.2 illustre les deux représentations d'une carte de contacts.

**Définition 2.2** *Carte de contacts*
*La carte de contacts d'une protéine $P$ est un graphe G=(V,E) avec $V = \{v_1, v_2, \ldots, v_n\}$. L'ensemble des sommets symbolise les résidus de la protéine (centrés en leurs $C\alpha$respectifs) et il existe une arête $e_{v_i,v_j}$ entre deux sommets $v_i, v_j \in V$ si et seulement si la distance entre les résidus correspondants respecte :*

$$dist(i, j) \leq \mu \;\; i, j \in P \; et \; un \; seuil \; réel \; fixé \; \mu \tag{2.18}$$

*La taille $|E|$ d'une carte de contacts correspond au nombre de contacts au sein de la protéine modélisée par G=(V,E).*

Cette représentation permet d'identifier rapidement et visuellement les résidus qui sont proches dans l'espace mais pas forcément dans la séquence. Cela offre donc un aperçu du repliement de la protéine.

La comparaison de deux cartes de contacts est appelée **recouvrement de cartes de contacts** (contact map overlap), CMO [39]). L'idée sous-jacente est que lorsque deux structures sont similaires, leurs cartes de contacts aussi. Par conséquent, mesurer la similarité de deux cartes de contacts permet d'estimer la similarité de deux protéines sans superposition. Le problème associé est le problème de maximisation du recouvrement de cartes de contacts, $maxCMO$, qui consiste à trouver le nombre maximal de contacts communs aux deux cartes. Ce problème est un problème NP-difficile [69]. Mais nous nous sommes intéressés à celle-ci car cette mesure s'est avérée efficace dans le cadre de la comparaison de structures protéiques [22, 4].

Dénombrer le nombre de contacts communs entre deux protéines permet de calculer différents scores de similarité/distance entre deux protéines et permet ainsi une comparaison globale de leurs structures.

La résolution du problème consiste à trouver l'alignement linéaire de résidus qui maximise le nombre de contacts communs entre les deux cartes de contacts.

### Apurva : un outil pour la résolution du problème maxCMO

Soient deux protéines $P_1, P_2$ et soit $G_1 = (V_1, E_1)\ et\ G_2 = V(V_2, E_2)$ deux cartes de contacts où $V_1$ (resp. $V_2$) correspond à l'ordonnancement linéaire des résidus de $P_1$. Il existe une arête $e_{v,w} \in E_1$ (resp. $E_2$) entre deux sommets $v, w \in V_1$ (resp. $V_2$) si et seulement si la distance entre les résidus correspondant est inférieure à un même seuil $\mu$.

Apurva est une méthode exacte de comparaison de cartes de contacts basée sur un algorithme de branch and bound qui, pour un temps de calcul donné, retourne le plus grand nombre de contacts communs ($LB(G_1, G_2)$) qu'il a trouvés au terme du temps imparti plus une estimation du nombre de contacts communs maximal ($UB(G_1, G_2)$) qu'il va pouvoir atteindre. LB est une borne inférieure au nombre maximal de contacts communs et UB en est une borne supérieure. Lorsqu'une limite de temps de calcul est fourni à Apurva, celui-ci se comporte comme une heuristique et retourne les bornes LB et UB à la fin du temps imparti. Si $LB, UB$ convergent, Apurva a terminé l'exploration et retourne $CMO(G_1, G_2)$, le nombre maximal de contacts communs entre les protéines $P_1, P_2$. $LB, UB\ et\ CMO$ respectent l'équation 3.2 suivante :

$$LB(G_1, G_2) \leq CMO(G_1, G_2) \leq UB(G_1, G_2) \tag{2.19}$$

### Scores de similarités basés sur la mesure $CMO$

La recherche et la mesure du nombre maximal de contacts communs ($CMO(P_1, P_2)$) est nécessaire mais insuffisante pour estimer la similarité de deux protéines. Il manque la relation avec les nombres de contacts $E_1, E_2$ présents au sein des protéines $P_1, P_2$.

Par conséquent trois scores sont proposés pour évaluer cette similarité :

1. Proportion de contacts communs sur le nombre moyen de contacts :

$$S_{sum} = \frac{2 \times CMO(G_1, G_2)}{|E_1| + |E_2|} \tag{2.20}$$

2. Proportion de contacts communs sur le nombre minimal de contacts :

$$S_{min} = \frac{CMO(G_1, G_2)}{min(|E_1|, |E_2|)} \tag{2.21}$$

3. Proportion de contacts communs sur le nombre maximal de contacts :

$$S_{max} = \frac{CMO(G_1, G_2)}{max(|E_1|, |E_2|)} \tag{2.22}$$

Ces scores ont pour point faible principal le fait qu'ils ne soient pas une mesure métrique, de plus nos expérimentations ont montré les limites qualitatives de ces scores. Nous avons donc chercher à estimer la distance entre deux protéines plutôt que la similarité.

**Scores de distances basés sur $CMO$**

Plusieurs distances ont été testées (voir l'article [115] pour plus de détails ) : $CMO(A, B)$ est le nombre de contacts communs entre les cartes de contacts A et B, et $|E_A|$ (respectivement $|E_B|$) est le nombre de contacts de la carte A (B).

$$\mathcal{D}_{sum} = 1 - \frac{2 \times CMO(A, B)}{|E_A| + |E_B|} \tag{2.23}$$

$$\mathcal{D}_{min} = 1 - \frac{CMO(A, B)}{min(|E_A|, |E_B|)} \tag{2.24}$$

$$\mathcal{D}_{max} = 1 - \frac{CMO(A, B)}{max(|E_A|, |E_B|)} \tag{2.25}$$

Les trois distances sont normalisées, cependant les distances $D_{min}$ et $D_{sum}$ ne satisfont pas l'inégalité triangulaire (preuve dans [115]), par conséquent ce ne sont pas des distances métriques et nous ne pouvons donc les utiliser. Cependant, la distance $D_{max}$ (équation 2.25) s'est avérée posséder la propriété d'être une distance métrique, ce qui induit la possibilité de caractériser l'espace des protéines à partir de cette distance. Par conséquent, cette nouvelle mesure peut être utilisée dans nos protocoles.

### 2.3.6 Discussion des scores

Cette section reflète la grande diversité des scores de similarités mise à disposition par la communauté, chacun d'eux espère capturer de manière pertinente la similarité entre deux structures protéiques. La littérature montre que tous y parviennent selon les objectifs initialement fixés. L'objectif de cette partie du mémoire étant la classification de structures nous vous renvoyons à la partie III pour une analyse de ces scores en fonction d'un alignement donné et leur intérêt dans la recherche de l'alignement optimal de deux structures. La grande difficulté de ces scores est de synthétiser toutes les informations relatives à la comparaison de deux structures dans une seule valeur qui,de prime abord, informe quant à la similarité des structures. Si nous prenons par exemple le TMscore, normalisé entre 0 et 1, nous savons qu'un TMscore supérieur à 0.5 reflète une similarité structurale importante : les deux structures ont un repliement semblable [118].

Théoriquement tous les scores cités précédemment pourraient être utilisés pour résoudre le problème de cette partie, à savoir comment intégrer une protéine dans une classification de protéines mais déjà certains peuvent être élagués comme les RMSD, utilisés sans relation avec leurs alignements, sont non-pertinents de par cette dépendance.

Les algorithmes utilisés en classification ont pour objectif de capter si deux structures partagent un même repliement de manière suffisamment précise. La vitesse est ici un point clé, au détriment de la précision, l'algorithme se doit d'être suffisamment précis pour différencier les grandes divergences mais il doit surtout être rapide. La classification de domaines requiert un score fiable et rapidement récupérable, raison pour laquelle les heuristiques sont souvent privilégiées. Les méthodes exactes comme DAST ou CMO, utilisées telles quelles, c'est-à-dire

en calculant avec exactitude leurs scores de similarité ne peuvent pas être utilisés dans le cadre de comparaisons à large échelle. La comparaison exacte d'une seule structure face à la classification entière se ferait en temps non raisonnable, inexploitable malgré la qualité de la méthode.

En conclusion, de nombreux outils permettent de mesurer la similarité de deux protéines, cette similarité sert de base aux classifications hiérarchiques et permet de regrouper des protéines dont les structures sont proches, donc dont les fonctions sont supposées proches. Dans la suite de cette thèse, nous montrons comment nous avons réussi à résoudre le problème de la résolution exacte d'une instance avec Apurva pour maintenir l'exactitude de la méthode tout en l'accélérant et en débouchant sur un protocole effectif en temps raisonnable. Pour démontrer l'efficacité de ces méthodes par rapport aux autres méthodes, nous avons comparé nos protocoles à une méthode de résolution du SFIP utilisant TMalign et le TMscore.

## 2.4 Samourai, un outil de mesure de scores à partir d'un alignement

Le grand nombre de scores existants et la nécessité de les comparer afin de mieux évaluer un alignement ont conduit à développer Samourai. Samourai est un outil qui, à partir d'une liste de résidus alignés (issue de n'importe quel outil d'alignement) et de deux structures (via leurs fichiers pdb) calcule une dizaine de scores. Il retourne aussi les valeurs correspondant à la meilleure matrice de superposition des deux structures.

Nous avons utiliser Samourai afin de mesurer et comparer tous ces scores. Non pour trouver le meilleur (ils n'ont pas les mêmes objectifs) mais pour trouver un outil qui va retourner les meilleurs scores pour une comparaison donnée. Samouraia été conçu afin de rendre possible la comparaison d'alignements issus d'outils différents et l'analyse de ces alignements grâce à un ensemble de scores non-exhaustif :

- RMSDc
- RMSDd
- RMSD100
- MI
- SI
- SAS
- Q_score
- Identité de séquence (%)
- TMscore (normalisé par les différentes protéines)
- Z-score
- S-score
- GSAS
- Similarité normalisée

Plusieurs études se sont intéressées à la pertinence de ces différents scores et d'autres ont vu le jour en essayant de trouver un moyen de synthétiser ces différents résultats. Est apparu entre autres le serveur CSA [116] qui, pour deux structures données, calcule des alignements selon quatre méthodes : CMO, DALIX, PAUL et MATRAS, et retourne différents scores pour chaque alignement.

Les outils d'alignement ayant chacun leur format de sortie, nous avons créé un format standardisé simple à parcourir et regroupant néanmoins les informations nécessaires et suffisantes à l'identification des paires d'atomes alignées.

Le principal inconvénient de Samouraiest qu'il mesure le score associé à un alignement donné. Cet alignement est lui-même issu d'un outil de comparaison qui optimise un score donné pour deux structures. Cependant, pour ces mêmes structures, un autre outil retournera un alignement optimisé selon un autre score. Par conséquent, les scores mesurés sont entièrement dépendants de l'alignement initial et ne reflètent pas la similarité optimale entre les structures. Les scores reflètent une similarité des structures selon l'alignement optimal associé à un score.

## 2.5 Résumé du chapitre

Cet état de l'art débute par la présentation de deux bases de données hiérarchisées de domaines structuraux dans lesquelles nous allons essayer d'introduire de manière automatique et sans erreur de nouveaux domaines. CATH et SCOP sont deux classifications hiérarchiques des domaines structuraux largement utilisées, elles concordent sur la classification d'une majorité de domaines ce qui tend à conforter leurs classifications respectives.

Ce chapitre présente également quelques uns des nombreux scores de similarité qu'il est possible de calculer pour entre deux structures et pose la question de déterminer le meilleur d'entre eux. Ces scores tendent à mesurer la similarité entre deux protéines selon différents critères structuraux ce qui les rend difficiles à comparer. Nous avons développé un outil qui évalue ces scores pour un alignement donné, à défaut de déterminer le meilleur score, il va nous être possible d'observer le comportement de ces scores en fonction de la paire de structures initialement comparée.

Nous avons également présenté Samourai, un petit outil de calcul de ces scores à partir d'un alignement donné.

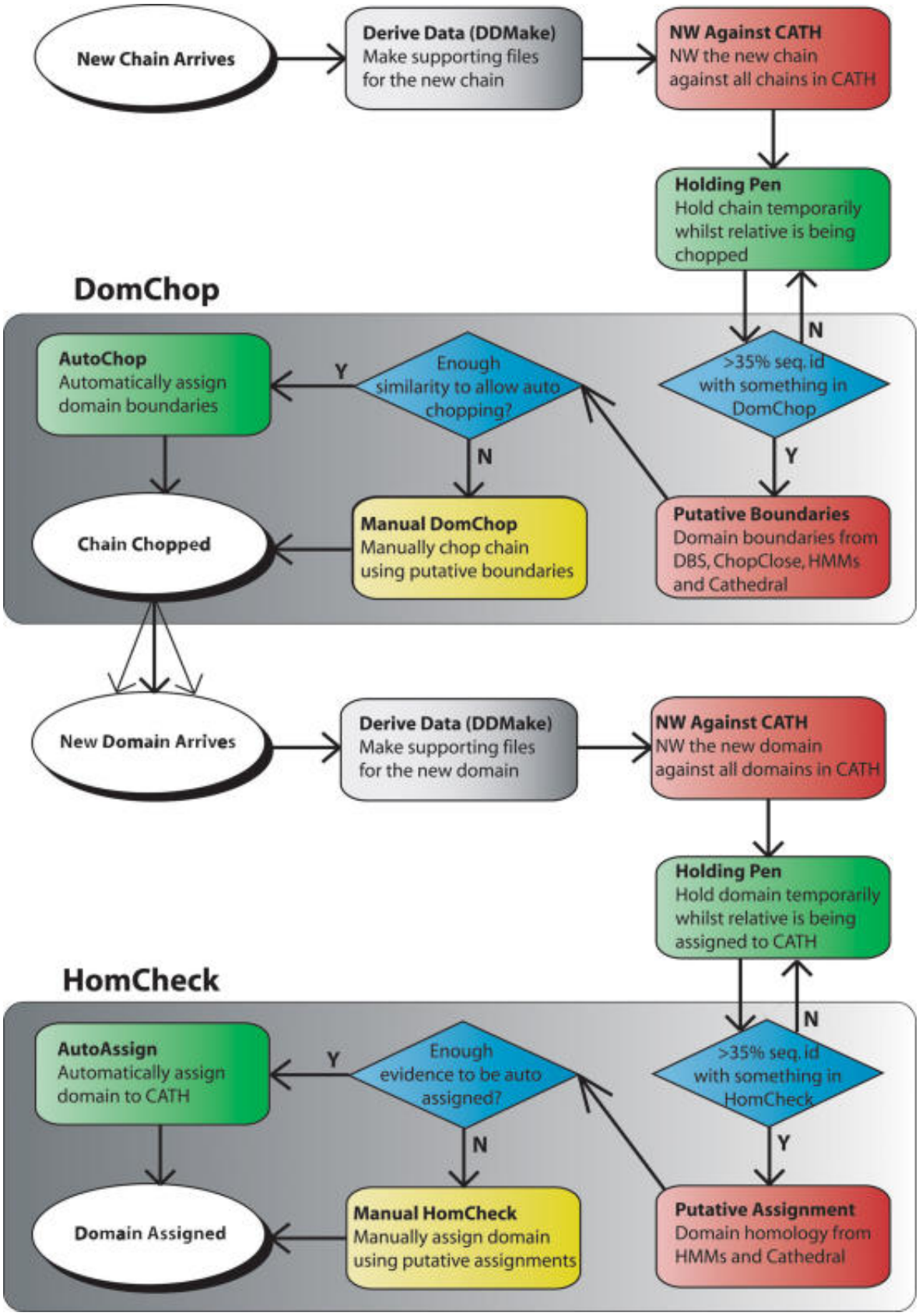



FIGURE 2.1 – Protocole d'identification de nouveaux domaines structuraux et assignation à une famille structurale.

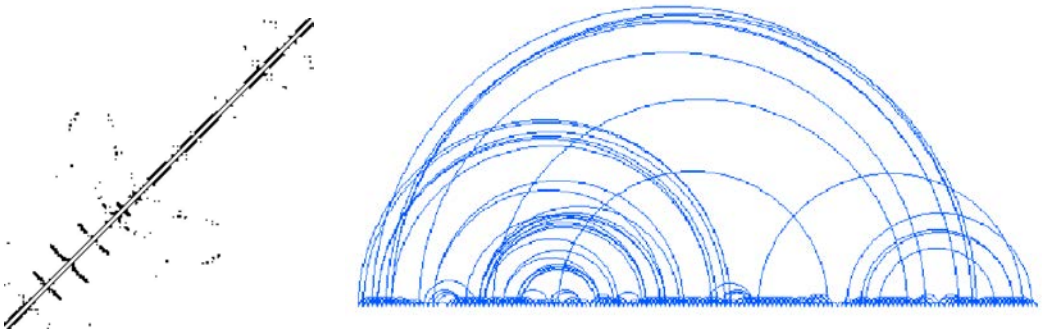


FIGURE 2.2 – Représentations d'une protéine par une carte de contact, (a) sous forme de matrice binaire avec 1 : contact, 0 sinon (à gauche) ou (b) sous forme de graphe (à droite.) Matrice calculée par `http://csa.project.cwi.nl/`

# Chapitre 3

# Résolution du problème d'identification de la super-famille structurale d'un domaine protéique

Ce chapitre est dédié à la présentation d'un protocole d'identification qui utilise les propriétés d'Apurva, un outil de mesure de similarités exact, pour identifier le plus proche voisin (Nearest Neighbour, noté NN) d'un domaine sans que la totalité des comparaisons soient effectuées jusqu'à optimalité. Nous présenterons tout d'abord le protocole initial, naïf, incluant le calcul de toutes les instances à la comparaison linéaire de leurs scores puis nous proposerons ici la notion de dominance entre instances, où une instance réfère à une comparaison entre deux domaines. Nous montrerons également précision de la méthode en terme de prédiction de superfamille structurale. La dominance permet d'élaguer un certain nombre d'instances qui, même une fois résolues, n'auront pas une similarité suffisante pour que leurs domaines cibles soient les plus proches voisins de la requête. La dominance diminue drastiquement le nombre de comparaisons à effectuer tout en conservant le caractère exact.

## 3.1 Méthode exhaustive ou one to all

Les scores utilisés dans ce chapitre sont commutatifs : $s(A, B) = s(B, A)$. Soit $q$ un domaine structural requête et $T = \{t_1, t_2, \ldots, t_n\}$ un ensemble de domaines structuraux cibles issus d'une base de données hiérarchique et $S : q \times T \rightarrow R^+$ une fonction de similarité qui associe à toute instance $(q, t_i)$, $t_i \in T$, un score de similarité $s(q, t_i)$. La recherche exhaustive, nommée méthode one to all, du plus proche voisin de la requête $q$ consiste à calculer pour toutes les instances $(q, t_i)$ le score de similarité associé puis de rechercher l'instance pour laquelle ce score est maximal. Cette méthode, décrite par l'algorithme 3.1, est utilisable avec n'importe quelle mesure de similarité s.

Parmi tous les scores de similarité proposés, nous utilisons CMO [38], via l'outil Apurva qui dénombre le nombre maximal de contacts communs entre deux domaines structuraux. A titre de comparaison nous avons utilisé le TMscore (via TMalign), qui est une mesure

**Algorithme 1** Méthode *one-to-all*, recherche du plus proche voisin (**NN**)

**Require:** $q, \mathcal{T} = \{t_1, ..., t_n\}$ ▷ domaine requête, ensemble de domaines cibles
  **for** $t_i \in \mathcal{T}$ **do**
    compute $s(q, t_i)$
  **end for**
  $NN = arg\ max(s(q, t_i)), argmax \in [0, 1]$

largement répandue. Ces deux méthodes nous permettent également d'observer les différences de comportements entre une méthode exacte et une heuristique.

### 3.1.1 Exemple d'application

La méthode a été testée sur le jeu de données SHREC'10 : le concours SHREC'10 *Protein Models Classification Track* (*SHape REtrieval Contest 10*). L'objectif de ce concours était d'observer l'efficacité des algorithmes de comparaison de structures 3D des protéines dans le cadre d'une classification [79].

Le jeu de données *SHREC'10* est composé de mille domaines structuraux protéiques issus de la classification hiérarchique CATH [88]. Chacun d'eux appartient à une famille protéique différente et les mille domaines se répartissent par groupe de dix dans cent superfamilles (niveau H). Ceci constitue la base de données à laquelle il faut ajouter cinquante domaines appartenant à cinquante des cent superfamilles qu'il va falloir retrouver.

L'objectif ici est donc de classer correctement les cinquante requêtes au sein de leur superfamille en utilisant la méthode présentée dans cette section. Deux points ont été particulièrement observés :

- le nombre de requêtes correctement insérées dans la base.
- le temps nécessaire à la réalisation de l'objectif.

Nous avons appliqué la méthode avec deux mesures de similarité : la mesure issue de CMO ( via Apurva, une méthode exacte) et le $TM - Score$ normalisé par la longueur de la requête (via l'outil TMalign, une heuristique).

TABLE 3.1 – Comparaison des performances des deux mesures en termes de temps de calculs nécessaires et de fiabilité des résultats

| Score | requêtes classées | requêtes correctement classées | Temps de calcul global |
|---|---|---|---|
| $\mathcal{S}_{sum}(CMO)$ | 50 | 46 | $> 1$ an |
| $TMscore$ | 50 | 48 | 1h42m58s |

Le taux de précision représente le pourcentage de requêtes qui ont été correctement classées par la méthode.

Le tableau 3.1 résume les deux expériences, on observe un taux de précision un peu plus faible mais correct pour la méthode basée sur le score exact, néanmoins chacune des méthodes présente une très bonne qualité de classification (92% et 96%). En revanche, les temps de calculs nécessaires divergent énormément. En utilisant l'heuristique, moins de deux

heures ont suffi à obtenir une classification, en revanche, il a fallu plus d'un an (en temps de calculs cumulés) à la méthode exacte pour le même protocole.

### 3.1.2 Analyse critique de la méthode et perspectives

Cette première méthode de classification présente le grand avantage de pouvoir être utilisée avec n'importe quelle mesure de similarité (issue d'une méthode heuristique ou exacte) mais présente quelques défauts.

- Chaque instance doit être résolue, cela est globalement rapide pour une heuristique mais l'utilisation d'une méthode de comparaison exacte requiert parfois des temps de calcul assez longs
- Si la similarité est déterminée par une heuristique, il n'y a pas de garantie que celle-ci ait produit la (ou l'une des) solution(s) optimale(s).
- Le nombre de données dans les bases ne cesse de croître, ainsi, rester sur une méthode obligeant une comparaison avec toute la base tend à devenir limitant, même avec une méthode très rapide.

Les tests ont montré que l'heuristique était beaucoup plus rapide que la méthode exacte, cela dit, elle présente les défauts d'une heuristique. La méthode exacte quant à elle est satisfaisante au niveau de la qualité de classification (mais néanmoins améliorable) mais présente des temps de calculs trop longs pour être qualifiés de raisonnables. Par conséquent, deux solutions sont envisageables : la première est de réduire les temps de calculs d'instances, la seconde de réduire le nombre d'instances à résoudre. La première solution dépend de l'algorithme d'Apurva que nous ne toucherons pas mais la seconde dépend de la méthode de recherche.

En conclusion, la recherche brutale du plus proche voisin est qualitativement acceptable mais la comparaison de notre méthode de comparaison CMO avec l'une des heuristiques les plus utilisées en a montré les limites temporelles.

## 3.2 Identification de superfamilles protéiques par dominance directe

L'un de nos objectifs a été de réduire le temps de calculs global tout en maintenant les performances de la méthode et son exactitude. Pour cela nous utilisons la propriété de bornes d'Apurva qui, pour chaque temps de comparaison, retourne deux valeurs encadrant avec certitude le nombre maximal de contacts communs entre les cartes de contacts considérées. Nous ajoutons également la notion de dominance entre instances.

### 3.2.1 Dominance exacte et dominance directe entre instances

Soient q, un domaine structural requête, $T = \{t_1, t_2, \ldots, t_n\}$ un ensemble de domaines structuraux et $S : q \times T \rightarrow \mathbb{R}^+$ une fonction de score. La dominance exacte a déjà été implicitement introduite dans la section précédente. En effet, on dit qu'une instance $(q, t_i)$ domine exactement une instance $(q, t_j)$ si $s(q, t_i) > s(q, t_j)$.

**Définition 3.1 (Dominance exacte)** *Soient deux protéines $t_i$ et $t_j$ ainsi qu'une protéine requête q et un score de similarité $s$. On dit que $t_i$ domine $t_j$ selon q si et seulement si :*

$$s(t_i, q) > s(t_j, q) \tag{3.1}$$

Le plus proche voisin correspondait donc à l'instance dominant exactement toutes les autres, ce quel que soit la fonction de score $S$.

A partir de maintenant, $S$ doit admettre pour toute instance $q, t_i$ deux bornes $\underline{s}(q, t_i)$ et $\overline{s}(q, t_i)$ telles que :

$$\underline{s}(q, t_i) \leq s(q, t_i) \leq \overline{s}(q, t_i) \tag{3.2}$$

Lorsque l'instance $q, t_i$ est résolue, on a :

$$\underline{s}(q, t_i) = s(q, t_i) = \overline{s}(q, t_i) \tag{3.3}$$

Les propriétés du score définies par les équations 3.2 et 3.3 permettent de définir la notion de dominance directe entre deux instances $(q, t_i), (q, t_j)$.

**Définition 3.2** *Dominance directe entre instances Soient $q, t_i, t_j$ trois structures et $s$ un score aux propriétés 3.2,3.3. L'instance $(q, t_i)$ domine l'instance $(q, t_j)$ si et seulement si :*

$$\underline{s}(q, t_i) \geq \overline{s}(q, t_j) \tag{3.4}$$

### 3.2.2 Insertion de la dominance dans le protocole d'identification des superfamilles

Le précédent protocole, exhaustif, nécessitait de résoudre l'ensemble des instances avant de pouvoir déterminer le plus proche voisin (NN). L'insertion de la dominance ôte ce besoin car permet d'élaguer au fur et à mesure les instances dominées. Ainsi, ne sont résolues que les instances pour lesquelles la dominance ne peut être établie. On introduit donc dans ce nouveau protocole un paramètre de temps $u$ qui va limiter la durée accordée à la résolution d'une instance, ainsi, à la fin du temps imparti, si l'instance n'est pas résolue, les bornes du score seront retournées et permettront l'élagage par dominance. Cela se fait par comparaison des bornes retournées.

L'utilisation de la dominance directe accélère la résolution du problème FIP de la manière suivante (décrite par l'algorithme 3.2.2) :

1. Toutes les instances $(q, t_i)$ sont initialisées dans une queue $I$. Le paramètre temps $u$ est affecté d'une faible valeur (2 secondes par exemple).
2. L'outil de comparaison calcule toutes les instances $(q, t_i)$, limitées par $u$ et retourne les scores $\underline{s}(q, t_i), \overline{s}(q, t_i)$ correspondants.
3. L'instance ayant le meilleur $\underline{s}(q, t_i)$ est considérée comme le plus proche voisin temporaire $(NN_t)$et sert de base pour la dominance directe.
4. Les instances $(q, t_j), t_j \in T$ telles que : $\overline{s}(q, t_j) < \underline{s}(NN_t)$ sont retirées de la queue
5. Si la queue ne contient plus qu'une instance, alors le plus proche voisin est trouvé, sinon les étapes 2, 3 et 4 sont répétées en augmentant graduellement $u$.

**Algorithme 2** Algorithme de recherche du plus proche voisin (**NN**) par dominance directe

**function** MAIN
  $q, T = \{t_1, ..., t_n\}$ ▷ domaine requête, ensemble de domaines cibles
  $u$ ▷ temps de calcul restreint
  $I = \{(q, t_1) : [\underline{s}(q, t_1), \overline{s}(q, t_1)], (q, t_2) : [\underline{s}(q, t_2), \overline{s}(q, t_2)], \ldots, (q, t_n) : [\underline{s}(q, t_n), \overline{s}(q, t_n)]\}$ ▷ Ensemble des instances initialisées
  **while** $|I| > 1$ **do**
    $\underline{s}(q, t_i), \overline{s}(q, t_i) = compute\_score(q, t_i, u) \forall I_i \in I$ ▷ Calcul des instances pour un temps $d$
    $apply\_dominance(I)$
    increase $d$
  **end while**
  **return** $NN_i = I[0]$
**end function**
**function** APPLY_DOMINANCE(I)
  $\underline{max} = max(\underline{s}(q, t_i)), (q, t_i) \in I$ ▷ recherche du meilleur score atteint
  $max \in [0, 1]$
  **for** $(q, t_i) \in I$ **do**
    **if** $\overline{s}(q, t_i) < \underline{max}$ **then**
      remove_instance(I) ▷ élagage de l'instance
    **end if**
  **end for**
**end function**

### 3.2.3 Résultats de la méthode sur le jeu de données SHREC'10

Nous avons à nouveau résolu le problème d'identification des superfamilles (SFIP) pour les 50 domaines structuraux de SHREC'10 en utilisant le nouveau protocole. Soit $G_i$ (respectivement $G_j$) la carte de contacts de la protéine $t_i$ (respectivement $t_j$). L'outil de comparaison est Apurva. Il permet de calculer le score $S_{sum}(G_i, G_j) = \frac{2 \times CMO(G_i,G_j)}{|E_i|+|E_j|}$ Le paramètre de temps a été initialisé à 2 secondes puis augmenté à 10 et 50 secondes. Cela donne les résultats du tableau 3.2 .

TABLE 3.2 – Résolution du problème FIP en utilisant la dominance directe avec Apurva. Pour chaque temps limite, le nombre d'instances calculées, le nombre d'instances dominées, le nombre d'instances restantes et le nombre de requêtes assignées au sein de la classification de SHREC.

| Temps limite (s) | instances | instances dominées | instances restantes | requêtes assignées |
|---|---|---|---|---|
| 2 | 50 000 | 49 721 | 229 | 43/50 |
| 10 | 229 | 227 | 2 | 48/50 |
| 50 | 2 | 2 | 0 | 50*/50 |

* Sur les 50 requêtes assignées, 46 l'ont été correctement. La méthode retourne quatre erreurs.

Le temps total de calcul s'élève à présent à moins de 29 heures (contre plus d'un an avec le protocole exhaustif). La qualité des résultats est maintenue au niveau de celle du protocole précédent (46 des cinquante requêtes ont été correctement prédites, tout comme précédemment).

### 3.2.4 Discussion, critique et pistes envisagées

Deux critiques majeures s'imposent concernant ce protocole. Tout d'abord même avec un élagage efficace, toutes les instances sont au minimum calculées une fois pour un laps de temps réduit. Ensuite la qualité de prédiction du protocole en utilisant le score de Apurva est bonne mais peut être améliorée. Par conséquent nous avons cherché d'autres mesures basées sur CMO.

Plusieurs pistes ont été évoquées, la première concerne l'élagage avant résolution de certaines instances en calculant un score dit trivial.

#### Borne triviale de CMO

Soient deux domaines structuraux modélisés sous forme de cartes de contacts par leurs graphes respectifs $G_i = (V_i, E_i)$ et $G_j = (V_j, E_j)$. $V$ étant l'ensemble des sommets (correspondant aux résidus du domaine) et $E$ l'ensemble des contacts répertoriés dans le domaine. Notre protocole se basant sur la mesure CMO et la maximisation de nombre de contacts communs ($CMO(G_i, G_j)$), nous savons d'emblée que, $CMO(G_i, G_j) \leq min(E_i, E_j)$. Le nombre de contacts communs ne peut effectivement pas excéder le nombre de contacts présents dans la plus petite structure. Donc, nous pouvons d'ores et déjà affecter une valeur à

la borne supérieure de $CMO$ : $UB$ que nous nommons $UB_{trivial}$.

$$UB_{trivial} = min(E_i, E_j) \tag{3.5}$$

Cette borne va pouvoir servir à calculer un score trivial :

$$\overline{s}(G_i, G_j)_{trivial} = \frac{2 \times UB_{trivial}}{|E_i| + |E_j|} \tag{3.6}$$

A partir de ce score, une fois le début des résolutions lancé, les instances non calculées mais bornées vont être potentiellement élaguées. Ainsi certaines instances ne seront pas du tout résolues mais à partir de ce score trivial seront élaguées.

#### Caractérisation de l'espace des protéines

Le problème d'identification des superfamilles suggère une classification initiale connue, invariable ou presque. Par conséquent, nous avons cherché à utiliser cette connaissance en calculant de prime abord des scores entre domaines d'un même groupe au sein de la classification puis cherché à déterminer si, à partir d'une mesure entre le domaine requête et l'un des domaines classés, il était possible de calculer sans passer par une résolution classique, le score de la requête avec un autre domaine en connaissant le score domaine-domaine. Cette recherche, décrite ultérieurement, a mené à une caractérisation de l'espace des protéines par une nouvelle mesure.

#### Recherche de scores et optimisation du protocole par vote

La volonté d'améliorer la qualité de prédiction du protocole ainsi que la recherche d'un score qui nous permettrait de caractériser l'espace des protéines a mené à l'utilisation de la distance $D_{max}$ définie dans le chapitre précédent. Ensuite, l'observation des résultats a montré que le plus proche voisin était souvent suivi par d'autres domaines issus de la même superfamille. Par conséquent, une optimisation de l'algorithme envisagé est non pas de stopper les calculs lorsque le plus proche voisin domine les autres mais lorsqu'il reste dans la queue uniquement des domaines du même groupe.

## 3.3 Identification de superfamilles protéiques par dominance directe et indirecte

Les travaux de cette section sont le fruit d'une collaboration avec Inken Wohlers, Gunnar Klau, Hristo Djidjev et Rumen Andonov. Les résultats sont extraits des publications produites [115]. Nous remplaçons ici l'ancien score de similarité $s(A, B)$ par la distance $D_{max}(A, B)$ prouvée métrique dans le chapitre 2 et nous nommons cette nouvelle mesure maxCMO. Cette caractéristique de la nouvelle mesure est fondamentale car elle permet d'appliquer l'inégalité triangulaire et ainsi évaluer la distance entre deux domaines à partir des distances

liées à un troisième domaine. De plus, pour nos exemples, nous utiliserons une classification hiérarchique, à plusieurs niveaux donc, en nous plaçant au niveau des superfamilles de domaines.

A la différence de la similarité, plus une distance est faible, plus les protéines sont proches. Ainsi les dominances entre deux instances (définitions 3.1 et 3.2) deviennent :

**Définition 3.3 (Dominance exacte basée sur la distance)** *Soient deux protéines $t_i$ et $t_j$ et une requête q. La protéine $t_i$ domine la protéine $t_j$ selon q si*

$$D_{max}(t_i, q) \leq D_{max}, t_j, q) \tag{3.7}$$

et

**Définition 3.4 (Dominance directe basée sur la distance)** *Soient deux protéines $t_i$ et $t_j$ et une requête q. La protéine $t_i$ domine la protéine $t_j$ selon q si*

$$\overline{d}(q, t_i) \leq \underline{d}(q, t_j) \tag{3.8}$$

De la formule de la distance $D_{max}$ (équation 2.25) sont déduites les distances issues des bornes :

$$\overline{d}(q, t_i) = 1 - \frac{LB(q, t_i)}{max(|E_q|, |E_{t_i}|)} \tag{3.9}$$

$\overline{d}(q, t_i)$ est la plus grande distance mesurée entre q et $t_i$, celle-ci ne peut que rester stable ou diminuer.

$$\underline{d}(q, t_i) = 1 - \frac{UB(q, t_i)}{max(|E_q|, |E_{t_i}|)} \tag{3.10}$$

A l'inverse, $\underline{d}(q, t_i)$ est la plus petite distance potentiellement mesurable entre q et $t_i$, celle-ci ne peut que rester stable ou croître.

### 3.3.1 Inégalité triangulaire entre domaines structuraux

Soient trois domaines structuraux p,q et r.

L'inégalité triangulaire dans un espace euclidien dit que, pour trois points A , et C, nous avons :

$$|AB - AC| \leq BC \leq AB + AC \tag{3.11}$$

Les propriétés métriques de notre mesure $D_{max}$ (notée $d$ pour la distance d'une instance) permettent d'appliquer cette inégalité triangulaire. Ainsi, en connaissant $d(G_p, G_q)$ et $d(G_p, G_r)$, il est possible de déterminer deux bornes $\underline{d}(G_q, G_r)$ et $\overline{d}(G_q, G_r)$ telles que :

$$\overline{d}(G_q, G_r) = d(G_p, G_q) + d(G_p, G_r) \tag{3.12}$$

$$\underline{d}(G_q, G_r) = ||d(G_p, G_q) - d(G_p, G_r)|| \tag{3.13}$$

Ces deux bornes triviales, intégrées au protocole présenté par la suite vont faciliter l'élagage.

### 3.3.2 Caractérisation de la classification, domaines représentants des superfamilles

Le fil conducteur des sections de ce mémoire est de minimiser le nombre d'instances à résoudre, ce pour une simple raison : le nombre important de domaines inclus dans les classifications. Soit $F$ une superfamille de domaines structuraux issue d'une classification $C$ quelconque et $d$ une mesure de distance métrique. $F$ peut être caractérisée par un domaine représentatif $R_F$ et un rayon $r_F$ définis comme suit :

**Définition 3.5 (Domaine représentatif et rayon)**

$$R_F = arg \min_{A \in F} \max_{B \in F} d(A, B) \tag{3.14}$$

$$r_F = \min_{A \in F} \max_{B \in F} d(A, B) \tag{3.15}$$

*$R_F$ est le domaine de $F$ le plus proche de tous les autres domaines de la famille et $r_f$ est la plus petite distance maximale entre les domaines, c'est-à-dire la distance entre $R_F$ et le domaine dont il est le plus éloigné.*

Si l'on représente visuellement une superfamille protéique, il s'agit d'une sphère dont le centre est $R_F$ et dont le rayon est la distance minimale qui contient tous les domaines au sein de la sphère (cf figure 3.1).

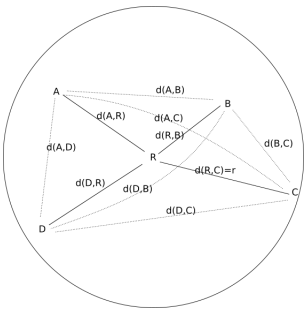



FIGURE 3.1 – Représentation d'une superfamille structurale avec son domaine représentatif R et le rayon r
Les distances entre le domaine représentatif et les autres domaines sont représentées par un trait continu tandis que les autres distances sont symbolisées par des pointillés.

L'identification de $R_F$ induit la résolution de toutes les paires $(p, q), p, q \in F$ de domaines de la superfamille. Par conséquent, toutes les distances correspondantes sont connues pour toutes les superfamilles avant le début du protocole SFIP.

### 3.3.3 Dominance indirecte entre instances

Soit $F$ une superfamille de représentant $R_F$, de rayon $r_F$ et $t_i$ un domaine appartenant à $F$. Soit également $q$ un domaine structural requête et $d$ une mesure de distance métrique. L'inégalité triangulaire permet d'estimer la distance $d(q, t_i)$ de la manière suivante :

$$|d(q, R_F) - d(R_F, t_i)| \leq d(q, t_i) \leq d(q, R_F) + d(R_F, t_i) \tag{3.16}$$

A partir de cette inégalité ainsi que de la dominance directe (basée ici sur la distance), on peut établir deux bornes nommées borne triangulaire supérieure et borne triangulaire inférieure. Ces bornes sont définies comme suit :

$$d^{\triangle}(q, t_i) = \overline{d}(q, R_F) + \overline{d}(R_F, t_i) \tag{3.17}$$

et

$$d_{\triangledown}(q, t_i) = max\{\underline{d}(q, R_F) - \overline{d}(R_F, t_i), \underline{d}(R_F, t_i) - \overline{d}(q, R_F)\} \tag{3.18}$$

Les bornes d'inégalité triangulaires suivent la même règle que les bornes sur la distance soit :

$$d_{\triangledown}(q, t_i) \leq d(q, t_i) \leq d^{\triangle}(q, t_i) \tag{3.19}$$

La grande force des bornes est qu'elles permettent d'appliquer une dominance entre deux instances. A partir de ces nouvelles bornes, nous pouvons calculer une nouvelle dominance : la *dominance indirecte entre instances*.

**Définition 3.6** *Dominance indirecte entre instances La protéine $t_i$ domine la protéine $t_j$ selon la protéine requête $q$ si $d^{\triangle}(q, t_i) < d_{\triangledown}(q, t_j)$.*

### 3.3.4 Protocole d'identification basé sur les bornes et la recherche des knn voisins

Ce protocole utilise toutes les notions de dominances précédemment vues afin de limiter le nombre d'instances à résoudre effectivement. Ici nous utilisons comme mesure de comparaison non plus un score de similarité mais le score $D_{max}$. De même, l'assignation d'un domaine structural à une superfamille s'effectuait selon la détermination du plus proche voisin (NN), à présent nous élargissons la méthode à la recherche des *k plus proches voisins* (kNN). Il est également nécessaire d'effectuer une analyse *ab initio* de la classification de référence afin de déterminer les domaines représentatifs et les rayons des superfamilles.

#### Analyse de la classification existante

Le protocole d'identification nécessite une unique étape de mesure des distances entre les domaines structuraux d'une même superfamille. A partir de ces distances sont déterminés les domaines représentatifs ainsi que les rayons des familles. Cette étape est nécessaire à l'utilisation de la dominance indirecte entre instances.

#### k plus proches voisins (kNN)

La recherche des kNN (*k Nearest neaighbours*) est une extension de la recherche du plus proche voisin. Il s'agit ici de déterminer les k domaines de la classification qui sont les plus proches du domaine requête. Une fois ces kNN identifiés, la requête est assignée à la superfamille majoritairement présente.

#### Protocole

Soit $q$ un domaine requête et $\mathcal{T}$ un ensemble de structures. Soient également $\mathcal{F} \in \mathcal{C}$ l'ensemble des superfamilles de la classification $\mathcal{C}$. Chaque domaine $t \in \mathcal{T}$ est associé à une unique famille $\mathcal{F}$. On suppose que pour chaque superfamille $F \in \mathcal{F}$, les distances exactes inter-domaines ont été mesurées et que le domaine représentant $R_F$ ainsi que le rayon $r_F$ ont été déterminés.

L'algorithme 3 (tiré de [115]) commence par estimer les bornes de chaque instance $(q, R_F)$. Puis les cibles $t$ sont triées dans deux files de priorité **LB** et **UB** par ordre croissant de la borne correspondante $(d_\triangledown, d^\triangle)$ calculée selon un temps donné. L'étape suivante est un élagage des structures $t$ dominées par $t_k^{UB}$. Si à la fin de cette étape, le nombre de structures restantes est égal à k alors l'algorithme retourne la superfamille majoritaire. Sinon le temps est incrémenté et le processus recommence au calcul des nouvelles bornes de l'instance $(q, R_F)$. Cela jusqu'à ce que toutes les bornes convergent ou que le nombre de structures dans la file **UB** ne soit plus modifié. La seconde étape majeure est une dominance directe, les instances $q, t$ sont calculées selon un paramètre temps croissant puis les instances dominées sont élaguées. A la fin du dernier tour (selon la dernière durée de comparaison utilisée -10 secondes ici-), les instances restantes, si la dominance n'est pas totale, sont triées selon LB et les kNN servent à l'assignation de la famille.

Les calculs de maxCMO pour les instances se font avec Apurva, à partir des bornes LB, UB qu'il retourne, les distances (exactes ou bornées) sont calculées. Nous avons ici limité le temps de calcul maximal à 10 secondes, cela implique que certaines instances peuvent ne pas être dominées et dans ce cas, la sélection des kNN est une heuristique puisque les kNN ne dominent pas l'ensemble des structures cibles $\mathcal{T}$.

### 3.3.5 Expérimentations

Nous avons testé ce nouveau protocole à l'aide de deux jeux de données issus de SCOPCATH[31]. SCOPCATH est le jeu de données consensuel des bases de données SCOP (1.75) [83] et CATH (3.2.0) [90]. Le jeu de données initial ne contient que des domaines avec un pourcentage d'identité de séquence inférieur à 50%. Cela équivaut à 6759 structures. La version étendue du jeu de données contient l'intégralité des structures consensuelles de SCOP et CATH, soit 67609 domaines structuraux. Ces domaines sont répartis dans 11 classes, 1348 superfamilles et 2480 familles (cf répartition dans le tableau 3.3). Les onze classes sont des spécifications des quatre classes usuelles ($\alpha$ principalement, $\beta$ principalement, $\alpha$ et $\beta$ avec feuillets $\beta$ parallèles, $\alpha$ et $\beta$ avec feuillets $\beta$ anti-parallèles) par des réarrangements des structures secondaires correspondant à des motifs caractéristiques dans les cartes de contacts.

**Algorithme 3** Solving the $k$-NN classification problem

1: $q$ ▷ Query structure.
2: $\mathcal{T}$ ▷ Set of target structures.
3: $R_F \ \forall F \in \mathcal{C}$ ▷ Family representatives.
4: $d(A, R_F) \ \forall A \in \mathcal{F}$ for all families $\mathcal{F} \in \mathcal{C}$ ▷ Distance from all family members to the respective representative.
5: $\underline{d}(q, R_F)$, $\overline{d}(q, R_F) \ \forall F \in \mathcal{C}$ ▷ Bounds on the distance from the query to the family representatives.
6: LB $\leftarrow \{(t, -\infty) | t \in \mathcal{T}\}$ ▷ Priority queue, which will hold the targets $t$ in the order of increasing lower bound distance $d_{\triangledown}(q, t)$ to the query.
7: UB $\leftarrow \{(t, \infty) | t \in \mathcal{T}\}$ ▷ Priority queue, which will hold the targets $t$ in the order of increasing upper bound distance $d^{\triangle}(q, t)$ to the query.
8: $t_k^{UB}$ ▷ A pointer to the $k$-th element in UB
9:
10: $\tau \leftarrow 1$ s ▷ Time limit for pairwise alignment.
11: **for** $\mathcal{F} \in \mathcal{C}$ **do**
12: FAM$[\mathcal{F}] \leftarrow |\{t \in \mathcal{T} : t$ belongs to family $\mathcal{F}\}|$ ▷ Number of family members.
13: **end for**
14: **while** $\exists R_{\mathcal{F}} : \underline{d}(q, R_{\mathcal{F}}) \neq \overline{d}(q, R_{\mathcal{F}})$ and $|\mathcal{T}|$ changes **do**
15: $\tau \leftarrow \tau \times 2$
16: **for** $\mathcal{F} \in \mathcal{C}$ with FAM$[\mathcal{F}] > 0$ **do**
17: Recompute $\underline{d}(q, R_F)$ and $\overline{d}(q, R_F)$ using time limit $\tau$
18: **for** $t \in \mathcal{F}$ **do**
19: Update priority of $t$ in LB to $d_{\triangledown}(q, t) = |\underline{d}(q, R_F) - d(R_F, t)|$. ▷ Bound from inverse triangle inequality.
20: Update priority of $t$ in UB to $d^{\triangle}(q, t) = \overline{d}(q, R_F) + d(R_F, t)$. ▷ Bound from triangle inequality.
21: **end for**
22: **end for**
23: ▷ Check for targets dominated by $t_k^{UB}$.
24: **for** target $t$ in $\mathcal{T}$ **do**
25: **if** $d_{\triangledown}(q, t) > d^{\triangle}(q, t_k^{UB})$ **then**
26: $\mathcal{T} \leftarrow \mathcal{T} \setminus t$
27: LB $\leftarrow$ LB$\backslash t$
28: UB $\leftarrow$ UB$\backslash t$
29: FAM$[\mathcal{F}] \leftarrow$ FAM$[\mathcal{F}] - 1$ where $\mathcal{F}$ is the family of $t$.
30: **end if**
31: **end for**
32: **if** $|\mathcal{T}| = k$ **then return** The majority superfamily membership $\mathcal{S}$ among $\mathcal{T}$.
33: **end if**
34: **end while**
35: Apply the dominance protocol for query $q$ and targets $t \in \mathcal{T}$ as described in [74]. (The quality of the bounds $\underline{d}(q, t)$ and $\overline{d}(q, t)$ are improved by stepwise incrementing $\tau$ within the given time limit. At each step, the direct dominance (def 3.4) is applied for the targets from the updated $\mathcal{T}$.)

algorithme issu de [115]

TABLE 3.3 – Répartition des domaines dans les classes pour les jeux de données SCOPCATH (str) et SCOPCATH étendu (ext) ainsi que le nombre de superfamilles(sup) et familles(fam) associés.

| **Class** | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** | **i** | **j** | **k** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| # str | 1195 | 1593 | 1774 | 1591 | 30 | 103 | 342 | 72 | 11 | 38 | 10 |
| # ext | 10,796 | 19,215 | 17,497 | 15,679 | 349 | 1006 | 2398 | 520 | 43 | 81 | 25 |
| # fam | 524 | 516 | 548 | 632 | 6 | 59 | 121 | 32 | 5 | 29 | 8 |
| # sup | 303 | 266 | 191 | 375 | 6 | 52 | 82 | 31 | 5 | 29 | 8 |

tableau issu de [115]

Afin de constituer les ensembles de domaines requêtes, nous avons sélectionné aléatoirement un domaine dans chaque famille constituée d'au moins six domaines. Par conséquent, l'ensemble de requêtes du petit jeu de données contient 236 domaines et celui du jeu de données étendu 1369 domaines.

### 3.3.6 Résultats

Pour analyser et comparer notre protocole, nous avons effectué un protocole one to all semblable à celui de la section 3.1. Seule différence, les kNN ont été sélectionnés et la requête a été assignée à la famille majoritaire. Nous avons testé le protocole pour des valeurs de k allant de 1 à 10. Cela a fait l'objet de deux articles [5] et [115] dont sont issus ces résultats.

#### Jeu de données SCOPCATH (236 requêtes, 6759 structures)

Le tableau 3.4 résume les résultats des assignations au niveau des familles. On observe que globalement le protocole permet de classer correctement plus de 85% des domaines requêtes. Ce pourcentage augmente avec la réduction du nombre de kNN. Cela tend à montrer qu'il est plus judicieux de se restreindre à la meilleure concordance plutôt qu'à un vote majoritaire. La ligne du tableau consacrée aux classifications exactes, c'est-à-dire aux requêtes pour lesquelles la kNN dominance a été totalement établie, que pour k = 1, dix requêtes ont été classées exactement mais dans la mauvaise famille. Ces exemples sont particulièrement intéressants car montre le pourcentage de limite de la méthode. L'explication de ces erreurs est soit au niveau de maxCMO qui ne permet pas d'estimer correctement la similarité entre les structures. La seconde option se situe au niveau du score $D_{max}$ qui, dans le cas de ces structures ne permettrait pas de correctement capter la distance entre les structures.

De même, TMalign qui est légèrement meilleur ne réussit pas à tout classer correctement.

Si l'on s'intéresse à présent aux requêtes classées correctement et de manière exacte, on remarque qu'une forte proportion d'assignations de requêtes est une approximation et non le fruit d'une dominance totale. De même le nombre d'égalité, c'est-à-dire les cas où deux ou plusieurs familles ont reçu le même nombre de votes et par conséquent l'assignation n'est pas possible, n'est pas négligeable pour les deux méthodes (maxCMO et TMalign).

TABLE 3.4 – Résumé des assignations des 236 requêtes de SCOPCATH pour maxCMO et TMalign.

Le tableau montre le nombre de requêtes correctement classées (correct), le nombre de cas où la dominance est totale (exact), les requêtes correctement et exactement classées (correct et exact) ainsi que les cas d'égalité (égalité).

| **k** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** |
|---|---|---|---|---|---|---|---|---|---|---|
| # correct | 210 | 211 | 213 | 213 | 214 | 217 | 217 | 219 | 213 | 224 |
| # exact | 117 | 143 | 156 | 165 | 188 | 206 | 204 | 211 | 209 | 234 |
| # correct et exact | 110 | 134 | 149 | 155 | 178 | 198 | 195 | 205 | 206 | 224 |
| # égalité | 10 | 9 | 11 | 8 | 10 | 10 | 10 | 10 | 20 | 0 |
| # TM-align correct | 219 | 220 | 220 | 225 | 225 | 228 | 226 | 227 | 226 | 228 |
| # TM-align égalité | 4 | 4 | 9 | 5 | 5 | 3 | 8 | 5 | 8 | 0 |

tableau issu de [115]

Pour cette expérimentation, nous avons paramétré le protocole pour lancer six fois la dominance indirecte (triangulaire) en augmentant le temps de calcul à chaque itération de 1 à 32 secondes CPU. Les même paramètres ont ensuite été utilisés pour la dominance directe. A cette étape la distance $d(q,t)$ est calculée directement. Nous avons voulu observer le nombre d'instances élaguées par chaque étape.

La figure 3.2 montre le pourcentage d'instances élaguées à chaque tour pour les dominances indirecte (triangulaire) et directe. Pour certaines requêtes, la dominance indirecte suffit à élaguer un grand nombre d'instances mais pour la majorité des requêtes il reste plus de 50 % des instances après cette étape.

La dominance directe est beaucoup plus efficace, mais également beaucoup plus coûteuse puisque les instances sont résolues (partiellement ou totalement) avec Apurva alors qu'il s'agit d'une simple opération mathématique dans le cas de la dominance indirecte. Pratiquement 100 % des instances sont élaguées à la fin de cette étape. Il reste néanmoins quelques cas où moins de 40 % des instances sont élaguées, la dominance n'est ici pas totale.

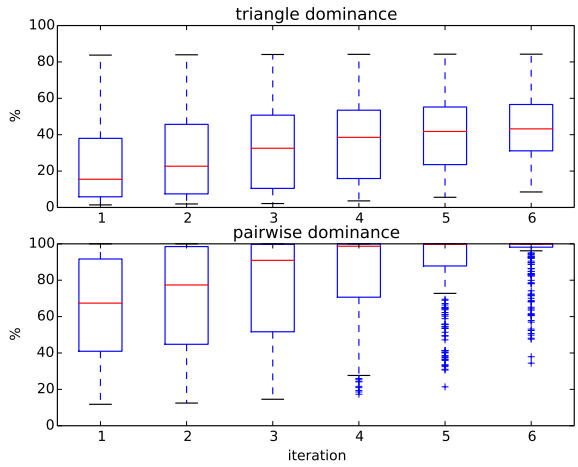



FIGURE 3.2 – Pourcentages d'instances élaguées lors des étapes de dominance indirecte (triangulaire) et directe (pairwise) pour les 236 requêtes du jeu de données SCOPCATH
figure issue de [115]

En conclusion, sur ce petit jeu de données, on remarque que la dominance indirecte permet d'élaguer un nombre non-négligeable d'instances et que la dominance directe, même partielle, permet d'obtenir plus de 85% de bonnes prédictions des requêtes.

### Jeu de données SCOPCATH étendu (1369 requêtes, 67 609 structures)

Les tests sur le jeu de données précédent ayant produits des résultats satisfaisants, nous avons testé le protocole sur un jeu de données dont la taille se rapproche de celles des classifications hiérarchiques existantes. Ici le nombre de requêtes correctement prédites est au minimum de 1303 sur 1369, soit 95% et de même la grande majorité des requêtes est classée via une dominance totale. Le tableau 3.5 résume les résultats du protocole. Les pourcentages de requêtes correctement assignées et de manière exacte (ou totale) ont augmenté par rapport au jeu de données précédent. Les temps de calculs varient entre 0.15 et 85.63 heures selon les requêtes (temps d'assignation), la durée moyenne étant de 3.8 heures.

TABLE 3.5 – Résumé des assignations des 1369 requêtes de SCOPCATH étendu pour maxCMO et TMalign.
Le tableau montre le nombre de requêtes correctement classées (correct), le nombre de cas où la dominance est totale (exact), les requêtes correctement et exactement classées (correct et exact) ainsi que les cas d'égalité (égalité). [115]

| **k** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** |
|---|---|---|---|---|---|---|---|---|---|---|
| # correct | 1303 | 1331 | 1334 | 1341 | 1341 | 1346 | 1344 | 1351 | 1348 | 1361 |
| # exact | 1120 | 1182 | 1228 | 1271 | 1286 | 1339 | 1341 | 1352 | 1347 | 1368 |
| # exact et correct | 1104 | 1166 | 1215 | 1257 | 1276 | 1329 | 1330 | 1341 | 1343 | 1360 |
| # égalité s | 35 | 5 | 12 | 6 | 11 | 7 | 9 | 3 | 17 | 0 |
| # TM-align correct | 1311 | 1347 | 1346 | 1350 | 1351 | 1354 | 1352 | 1353 | 1351 | 1361 |
| # TM-align égalités | 39 | 4 | 7 | 4 | 6 | 4 | 4 | 5 | 15 | 0 |

On remarque que pour k = 1, TMalign et notre protocole retournent le même nombre de prédictions correctes, de plus une seule requête est classée par approximation (mais correctement). Cela signifie qu'il existe 8 requêtes classées après une dominance totale mais faussement.

La figure 3.3 présente l'élagage des instances selon les étapes de dominance indirecte (triangulaire) et directe. Comme précédemment, une bonne partie (60% ici) des structures sont élaguées durant la première étape. De plus, le premier tour de la dominance directe permet d'obtenir une dominance totale pour plus de 70% des requêtes.

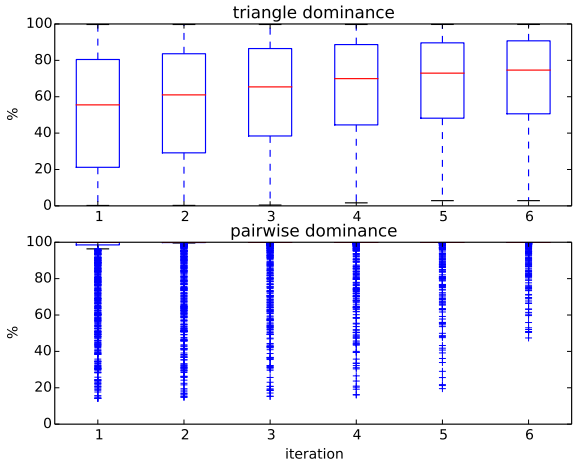



FIGURE 3.3 – Pourcentages d'instances élaguées lors des étapes de dominance indirecte (triangulaire) et directe (pairwise) pour les 1369 requêtes du jeu de données SCOPCATH étendu
figure issue de [115]

## 3.4 Discussion, perspectives, travaux en cours

Les expérimentations sur les jeux de données SCOPCATH et SCOPCATH étendu ont retourné de bons résultats avec la méthode basée sur maxCMO que ce soit en terme de qualité de prédiction ou de performances. Cependant, on a observé que, que ce soit pour TMalign ou maxCMO, l'utilisation des kNN diminuait la précision des résultats. L'une des explications est l'absence de pondération dans le vote de chacun des kNN, la famille du plus proche domaine est considéré de la même manière que celle du kième. Or on note que lorsque k=1, les prédictions sont meilleures. Par conséquent il serait intéressant d'envisager de pondérer les votes pour donner une importance plus ou moins grande selon la position du domaine dans la liste.

Un point faible ici est la différence de précision avec TMalign, les résultats ont montré que pour le protocole actuel (deux étapes de dominance avec 6 tours) la précision était légèrement moindre que celle de l'heuristique. Cependant le nombre de requêtes dont la dominance est totale (dans ce cas le protocole est exact) n'est pas maximal. Ce nombre peut être augmenté en modifiant le nombre de tours autorisés (et le temps de calcul par instance augmenté donc) notamment dans la seconde étape (dominance directe). En effet l'une des causes de mauvaise assignation est l'erreur induite par l'approximation, en dominant plus largement, voire totalement, on réduit le nombre de requêtes faussement assignées. Accorder plus de temps de calcul à chaque instance augmente certes la durée potentielle de l'assignation mais les bornes s'affinent et donc la dominance aussi.

**Apprentissage** Une amélioration supplémentaire est possible grâce à la nouvelle version d'Apurva qui accepte des bornes en paramètres d'entrée [1]. Jusqu'à présent, à chaque résolution d'instance pour un temps t, Apurva débutait la comparaison à 0 et retournait les valeurs de bornes après t secondes. Ces bornes vont être conservées d'une comparaison à l'autre (pour une même instance avec un temps de calcul différent) et ainsi éviter de recommencer la comparaison du début. Cet apprentissage des bornes est un gain de temps non négligeable et ainsi réduire la durée globale du protocole.

La réduction du temps de résolution des instances l'une des deux méthodes d'optimisation du protocole. La seconde méthode est la réduction du nombre d'instances à comparer. Ce fut déjà le cas avec l'utilisation de la dominance indirecte qui élague un certains nombre d'instances, nous envisageons à présent d'utiliser la *dominance entre superfamilles*.

### 3.4.1 Dominance entre superfamilles

**Définition 3.7** *Dominance entre superfamilles Soient $F(R_F, r_F)$ et $G(R_G, r_G)$ deux superfamilles avec $R_F, r_F$ (respectivement $R_G, r_G$) le domaine représentatif et le rayon de la famille $F$ (resp. $G$) et $q$ un domaine requête. Si $\overline{d}(q, R_F) + r_F < \underline{d}(q, R_G) - r_G$, alors la famille $F$ domine la famille $G$.*

La dominance entre superfamilles permet d'élaguer des superfamilles et ainsi réduire le

1. Implémentation par Noël Malod-dognin

nombre d'instances à effectuer. Les premiers travaux ont consisté à étudier le chevauchement des superfamilles avec en première expérimentation les familles du jeu de données SKOLNICK [67] et leur dispersion dans l'espace. [2] Le jeu de données SKOLNICK est constitué de 40 domaines structuraux répartis en 5 familles 7.2. Pour chaque famille, les domaines représentatifs ainsi que les rayons ont été mesurés puis le chevauchement des familles a été observé.

TABLE 3.6 – Jeu de données SKOLNICK

| | Familles SCOP | Taille | Protèins |
|---|---|---|---|
| 1 | CheY-related | 120-130 | 1b00, 1dbw, 1nat, 1ntr, 3chy 1qmp(A,B,C,D), 4tmy(A,B) |
| 2 | Plastocyanin /azurin-like | 97-105 | 1baw, 1byo(A,B), 1kdi, 1nin 1pla, 2b3i, 2pcy, 2plt |
| 3 | Triosephosphate isomerase (TIM) | 243-256 | 1amk, 1aw2, 1b9b, 1btm, 1hti 1tmh, 1tre, 1tri, 1ydv, 3ypi, 8tim |
| 4 | Ferritin | 158-191 | 1b71, 1bcf, 1dps, 1fha, 1ier, 1rcd |
| 5 | Fungal ribonucleases | 104 | 1rn1(A,B,C) |

Soient deux familles $F, G$ avec $R_F, r_F$ et $R_G, r_G$ les domaines représentatifs et les rayons associés. Les familles se chevauchent si la distance euclidienne entre les domaines représentatifs est inférieure à la somme des rayons. Le jeu de données de Skolnick est assez "simple" dans le sens où les familles choisies sont très éloignées les une des autres, par conséquent il n'est pas étonnant que celles-ci ne se chevauchent pas. En revanche, lorsque l'on applique cette comparaison à des jeux de données plus larges comme PROTEUS[4], composé de 300 domaines structuraux issus de SCOP (séparées en 30 familles de 10 domaines), la majorité des familles se chevauchent. Néanmoins, lors de ces expériences, en moyenne 12 familles sur 30 étaient élaguées . Cela montre que la dominance de familles a un potentiel non négligeable de réduction du temps d'assignation d'un domaine à une famille.

### 3.4.2 Combinaison des différentes dominances dans un seul protocole

L'ensemble des différents protocoles nous permet d'émettre l'hypothèse que, toutes dominances combinées, il est possible de créer un protocole automatique d'assignation d'un protocole à une famille/superfamille fiable.

Nos différentes expérimentations ont porté sur les deux niveaux de classifications et peuvent être substitués dans le protocole. Nos premières analyses (SHREC'10) inséraient des structures au sein de superfamilles, SCOPCATH des familles.

Notre nouveau protocole utilise la distance $D_{max}$, il devrait être potentiellement plus lent que celui de SCOPCATH car nous allons jusqu'à la dominance totale mais nous espérons

2. Travaux réalisés par Nada ABASSI durant son stage au sein de l'équipe Genscale

une meilleure qualité de prédiction. De plus ici, seul le NN sera utilisé. Le protocole assigne actuellement un domaine à une superfamille selon les étapes suivantes :

1. Etape préliminaire : mesure des distances entre domaines et détermination des domaines représentatifs et des rayons pour chaque superfamille ;
2. Mesure des distances exactes entre le domaine requête et tous les domaines représentatifs ;
3. Application de la dominance entre familles, élagage des superfamilles dominées ;
4. Application de la dominance indirecte entre instance (même utilisation que lors du protocole de SCOPCATH soit 6 tours (1 à 32 secondes CPU) ;
5. Application de la dominance directe entre instances jusqu'à atteindre une dominance totale (exacte)

La dominance de chaque étape est dite totale si, dans la liste des instances restantes, tous les domaines appartiennent à la même superfamille ou s'il ne reste que le NN. De plus, nous allons utiliser ce nouveau protocole avec la nouvelle version d'Apurva dont l'utilisation de bornes en paramètres devrait permettre de réduire les temps de calculs des instances.

Nous allons tester ce protocole sur une version étendue de SHREC'10 [79]. SHREC'10 est composé de 1000 domaines tirés de CATH et classés dans 100 superfamilles différentes (à raison de 10 domaines par familles). 50 domaines, de 50 superfamilles différentes parmi les 100 (mais hors des 1000 domaines) font office de requêtes. Nous allons étendre ce jeu de données en récupérant tous les domaines de chaque superfamille desquels nous allons retirer une structure qui fera office de requête. Nous appliquerons ensuite ce nouveau protocole sur l'ensemble des requêtes.

### 3.4.3 Perspectives : analyse des bêtes noires

Les *bêtes noires* sont les domaines structuraux requêtes qui ne sont pas correctement prédits. Dans le jeu de données SHREC'10, elles étaient au nombre de 4 (1TTEA02, 1WWJA00, 1JFTA01 et 3BIOA02). Elles résistaient à tous les protocoles, scores CMO et autres jeux de paramètres. La conclusion était que l'utilisation des cartes de contacts ne permettait pas de correctement prédire la superfamille de ces domaines. TMalign bloquait également, et l'utilisation d'un outil plus local, DAST, qui possède aussi des bornes mais sur le nombre de paires de résidus alignés, ne prédisait que trois des 4 bêtes noires. Pour la dernière, 3BIOA02, nous avions dû consulté Alexei Murzin qui nous avait judicieusement recommandé d'ajouter un domaine dans notre jeu de données pour parvenir à une classification. Ces quatre domaines possédaient un point commun : lorsque l'on mesurait un RMSDc normalisé basé sur l'alignement des domaines avec leurs NN respectifs prédits par le protocole, celui-ci était bien plus élevé que pour les autres paires domaine requête-NN (figure 3.4).

Connaissant ces limites, l'une des pistes envisagées est de combiner un protocole classique avec un autre outil de comparaison comme DAST avec des bornes ou sans bornes. Mais, dans le cas d'un outil sans bornes, cela revient à comparer la requête à toute la base, donc il est nécessaire de poursuivre la réflexion pour trouver un moyen de limiter le nombre d'instances.

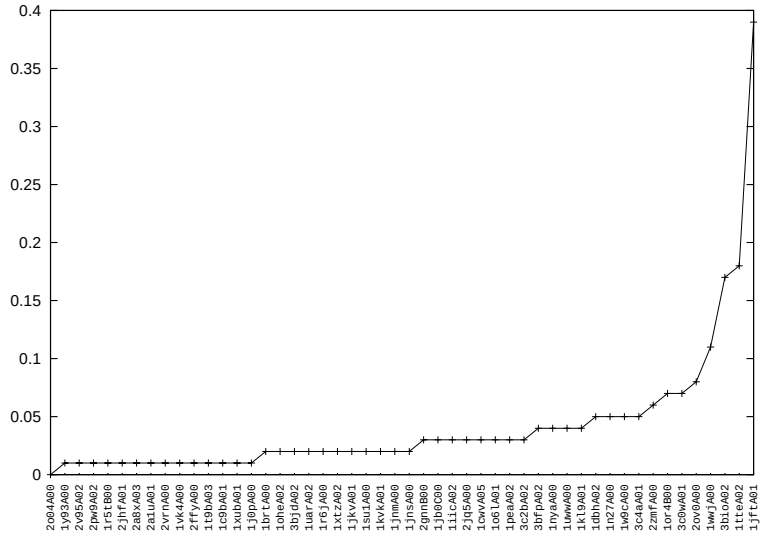



FIGURE 3.4 – Résultats de SHREC'10 en utilisant la dominance directe et Apurva, pour chaque couple (requête,NN), noté $(q, NN)$ : $NRMSDc(q, NN) = \frac{RMSDc}{len(q)}$, $NRMSDc \in [0, 1]$

En abscisse sont listées les domaines structuraux requêtes et en ordonnée les valeurs de RMSDc normalisées (NRMSDc).

## 3.5 Conclusion

Nos expérimentations ont montré qu'utiliser la mesure maxCMO avec un protocole exact et une mesure de distance métrique permettrait de correctement prédire la famille d'un domaine structural de manière automatique et efficace. L'introduction des dominances dans les protocoles diminue le temps global moyen nécessaire pour qu'un domaine requête soit assigné à une superfamille. Nos évaluations sur des jeux de données dont la taille se rapproche de celles des classifications hiérarchiques existantes a servit à estimer la pertinence d'un tel protocole en conditions réelles. Les études sur le nombre de plus proches voisins (kNN) tendent à montrer qu'il vaut mieux choisir le plus proche voisin. Mais les cas de mauvaises prédictions persistent et nécessitent une étude plus poussée pour déterminer le point de faiblesse. En conclusion les résultats obtenus montrent que la version actuelle du protocole est assez fiable et rapide pour être utilisée par la communauté. Il reste encore des améliorations à intégrer, notamment pour détecter automatiquement qu'un domaine a été mal assigné. A défaut de connaître la bonne superfamille, cela permettra au moins de connaître le cas d'erreur et de soumettre le domaine à d'autres protocoles utilisant une autre méthode de comparaison de structures.

## 3.6 Résumé du chapitre

Ce chapitre fut dédié à la présentation des notions de dominances entre instances, une instance étant la comparaison de deux structures, dans un but d'assignation d'une structure à une famille/superfamille protéique. Le protocole standard d'assignation nécessite de comparer un domaine structural requête à tous les domaines de la classification. Les dominances permettent de restreindre ce nombre de comparaisons tout en garantissant de n'élaguer que des domaines ne pouvant prétendre à influer sur l'assignation. Les méthodes initiales sont basées sur un score de similarité mais nous avons introduit une mesure de distance, qui est une métrique et permet ainsi d'utiliser des relations fortes comme l'inégalité triangulaire. Elle permet aussi de caractériser l'espace des protéines.

Nous avons présenté différents protocoles intégrant les différentes notions de dominances exactes, directes et indirectes entre instances ainsi que la dominance entre superfamilles protéiques.

# Troisième partie

# Comparaison fine de structures protéiques et alignements structuraux

# Introduction

Comparer deux structures protéiques, c'est avant tout comparer deux objets tridimensionnels. Il s'agit donc de comparer deux ensembles de points dans l'espace. Cependant, lorsque l'on s'intéresse aux protéines, plus exactement à leurs fonctions, il faut ajouter une étude de la biochimie des structures afin de parvenir à une analyse complète. C'est pourquoi le cœur de cette thèse est composé de deux aspects représentés par deux modules : Shinobi et Ninjas. Shinobi modélise une question biologique dans un graphe d'alignement tandis que Ninjas compare deux ensembles de points dans l'espace tridimensionnel (3D) modélisés par un graphe d'alignement.

Ensuite se pose la question de la pertinence de l'alignement obtenu. Dans la partie précédente nous cherchions à mesurer, à capter une similarité entre deux structures. Dans cette partie nous cherchons à comprendre les alignements, la manière dont les outils captent les similarités et les traduisent dans leurs alignements. Nous verrons notamment que deux outils retournant des alignements différents peuvent avoir capté la même similarité mais l'exprimer différemment.

Cette partie se divise en plusieurs chapitres : un état de l'art ayant permis de choisir des outils portant sur différents aspects de la comparaison de structures, quatre chapitres portés sur nos méthodes : la recherche d'éléments similaires ou divergents entre deux protéines correspondants à des alignements structuraux ainsi qu'une discussion générale sur la qualité des alignements et les poursuites d'études à effectuer pour obtenir une analyse globale.

Nous avons utilisé un outil présenté dans la section précédente : Samourai, qui permet de calculer de nombreux scores pour un alignement donné. Cela nous permet de comparer deux alignements mais également de noter les divergences entre les scores. Comme écrit précédemment nous présenterons ici nos outils ayant permis de tester nos différentes hypothèses et méthodes.

- Comment modéliser la comparaison de deux structures en faisant ressortir des caractéristiques à la fois géométriques et physico-chimiques ?
  Nous avons choisi d'utiliser le formalisme des graphes qui permet d'ajouter au modèle de base (un graphe étant initialement un ensemble de sommets et d'arêtes) des propriétés de tout type.
  Shinobi ("l'homme ninja") va modéliser les structures protéiques dans un graphe en tenant compte des informations intégrées issues de la séquence et des structures secondaires pour optimiser le graphe créé et répondre au mieux à la problématique.
- Comment détecter des éléments communs à deux structures protéiques ?

Le graphe créé contient ces informations qui, après détection, constituent des alignements structuraux. Ninjas, qui parcourt le graphe créé par Shinobi et renvoie des pseudocliques. Ces pseudocliques correspondent à de nombreux alignements pour une comparaison donnée, ce qui soulève la question du meilleur alignement et des alignements alternatifs.

- Comment détecter les protéines solénoïdes ? Comment trouver un motif structural qui, répété x fois, recouvre une protéine ?
  Toujours avec une modélisation par un graphe, représentant non plus une protéine comparée à une autre mais face à elle-même, nous recherchons ici des ensembles d'alignements structuraux tous similaires. Dans un graphe, ils correspondent à une clique, non plus à une pseudoclique. Kunoichi ("la femme ninja") utilise des principes des deux outils précédents pour rechercher des répétitions internes au sein des structures protéiques
- Comment trouver des divergences au sein de structures fortement similaires ?
  L'alignement des structures n'est plus ici la finalité mais le point de départ de l'analyse. A partir de la superposition en 3D des structures, nous cherchons à identifier et caractériser toutes les fonctions biochimiques d'une structure à l'autre. Nous avons implémenté Daijinushi("l'écuyer") sous forme d'extension Chimera.

# Chapitre 4

# Outils pour l'alignement 3D de deux structures

## 4.1 Introduction

L'analyse locale, par comparaison à la comparaison globale de protéines, se focalise sur les sous-structures similaires mais surtout sur la compréhension de ces similarités avec une étude affinée. Là où une comparaison globale va conclure que deux protéines sont similaires à 70%, l'analyse locale va identifier les sous-structures impliquées précisément. De plus, les comparaisons locales ont également pour but la détection des sous-structures « importantes » comme les sites de liaisons ou les sites catalytiques des protéines. Enfin, ces analyses ont pour but la mise en évidence des mouvements de structures, qu'ils soient dûs à des charnières ou bien à des permutations de structures. Certaines analyses ont également été dédiées à la recherche des répétitions internes chez les protéines. Les sections suivantes présentent les principaux algorithmes d'alignements ainsi qu'une discussion autour des modèles utilisés pour représenter les protéines. Il existe de nombreux outils d'alignements structuraux, et presque chaque outil a sa propre fonction objectif. Les premiers ont maintenant plus de vingt ans et de nouveaux apparaissent régulièrement sans qu'aucun ne devienne la référence dans la communauté scientifique. C'est à la fois un bon point car la recherche avance toujours plus, les alignements réputés difficiles mettent à rude épreuve les outils qui évoluent et l'on s'aperçoit qu'à défaut d'avoir un outil capable de résoudre l'ensemble des cas, beaucoup d'outils résolvent chacun un cas. Une avancée notable dans l'évolution des algorithmes est la prise en considération de l'indépendance de la structure par rapport à la séquence : les structures étaient initialement considérées linéairement et qu'il suffisait de trouver le meilleur alignement structural en « suivant » l'ordre d'apparition des acides aminés le long de la séquence (de l'extrémité N-terminale à l'extrémité C-terminale). Ce postulat a mené à ce que l'on a nommé les alignements ordre-dépendants ou **alignements séquentiels** . La découverte des permutations circulaires au sein de certaines protéines a mis à mal ce postulat et la plupart des outils d'alignements structuraux séquentiels (notamment ceux qui contraignent leur modèle avec ce critère de suivi de la séquence et qui ont optimisé leurs algorithmes en fonction). Car si les nouveaux algorithmes, ordre-indépendants ou **non-séquentiels** traitent

indifféremment les deux cas, les algorithmes séquentiels ne retournent qu'un alignement partiel correspondant à la plus longue sous-structure linéaire trouvée.

Les alignements structuraux ne prennent généralement en considération que le carbone central ($C\alpha$) de chaque acide aminé, réduisant les protéines à des ensembles de points ordonnés. Initialement, chaque résidu d'une protéine peut être aligné avec n'importe quel autre résidu de la seconde protéine mais certains outils réduisent ces possibilités en se basant sur des critères d'appariements basés sur les structures secondaires ou des cartes de contacts ou encore des motifs d'interactions [38].

Ce chapitre présente les grandes catégories d'algorithmes de comparaison d'une paire de structures de manière non-exhaustive avec leurs caractéristiques, leurs forces et leurs faiblesses. Nous avons sélectionné et décrit plus en détail quelques outils représentatifs : Apurva (présenté dans la partie précédente), TMalign [126],MICAN [80], FlexSnap [102] et PROBIS [63] et observer leurs résultats sur différentes instances issues de la littérature. Ces outils servent de points de comparaison afin de mesurer les performances de nos propres outils. Ces méthodes ont des scores différents, présentés en section 2.3 si bien qu'il est difficile de comparer leurs résultats.

## 4.2 Alignements séquentiels basés sur la minimisation des différences de distances intra-atomiques

Les outils pour l'alignement 3D de protéines cherchent à créer un alignement de taille maximale avec une erreur de déviation basée sur les distances minimale (ou sous un seuil selon les méthodes). La modélisation d'une protéine en carte de contacts consiste à créer une matrice de distances carrée (4.1) dont les lignes/colonnes correspondent aux résidus de la protéine et où une case i,j a pour valeur la distance entre ces résidus. De cette matrice de distance sont extraites toutes les distances inférieures à un seuil ($\mu$Å), les résidus correspondants sont dits en contact. L'une des représentations les plus courantes est sous forme de graphe (figure 4.1). La protéine est modélisée par sa structure primaire (séquence d'acides aminés) et il existe une arête entre deux résidus si ceux-ci sont en contact.

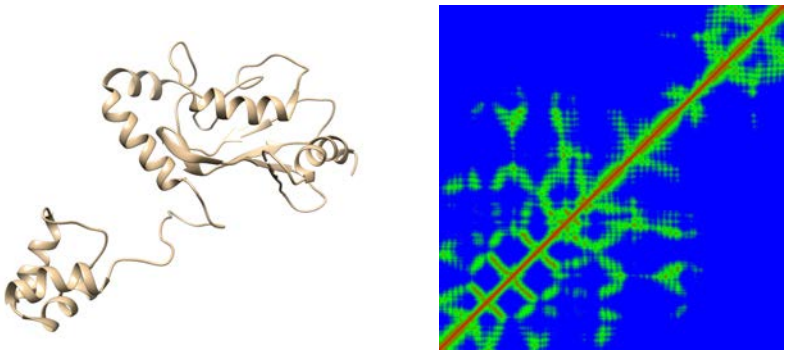


FIGURE 4.1 – La protéine 1TTE(A) (à g.), et sa matrice de distances associée (à d.) Matrice calculée par `http://csa.project.cwi.nl/`, le rouge dénote des résidus très proches, à l'inverse le bleu signale des résidus éloignés

Les principaux outils basés sur les matrices de distances sont DALI [51], Matalign [12], ou encore DAST. DALI, Distance mAtrix aLIgnment [51] est l'une des heuristiques les plus utilisées et dont il existe une version exacte (DALIX [116]). DALI compare des matrices de distances en les décomposant en sous matrices de taille fixe (contact patterns). Les sous-matrices sont comparées et évaluées, et les paires de résidus validées sont stockées dans l'alignement. Enfin les paires sont triées selon l'ordre de la séquence primaire et correspondent au résultat.

Le problème de recouvrement de cartes de contacts consiste à chercher un alignement ordre-dépendant (pas de croisements ) de résidus issus de deux protéines A et B considérés comme équivalents tels que le nombre de contacts communs aux deux protéines est maximisé. Caprara et collègues [22] ont développé plusieurs algorithmes basés sur des méthodes de programmation linéaires pour résoudre ce problème, de même il existe des algorithmes tels que PAUL ou encore Apurva pour résoudre ce problème.

L'un des points forts de ces algorithmes est l'affranchissement de la superposition, le calcul de la superposition optimale n'est pas nécessaire puisque les distances inter-résidus

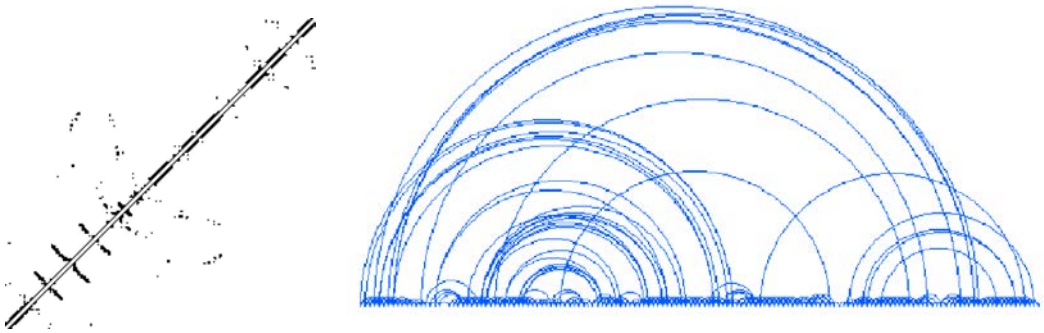


FIGURE 4.2 – Représentations d'une protéine par une carte de contact, (a) sous forme de matrice binaire avec 1 : contact, 0 sinon (à gauche) ou (b) sous forme de graphe (à droite.) Matrice calculée par `http://csa.project.cwi.nl/`

ne changent pas. De plus, un effet de bord est la prise en compte implicite de la flexibilité des protéines. Si l'on prend l'exemple des charnières avec Apurva, ces larges changements de conformations, ils se modélisent au sein de la carte de contact par un nombre de contacts différent au niveau de la charnière mais le reste des distances dans les autres parties des protéines ne varie pas. Donc, Apurva va aligner les différentes parties des protéines et retourner un alignement global qui intégrera implicitement la charnière. L'inconvénient est que ces outils ne sont pas capables de détecter qu'ils intègrent une charnière, ils retournent un alignement qui, mesuré avec des scores classiques comme le RMSDc, renvoie de mauvais scores comme le montre l'exemple 4.1.

Les résultats des quatre outils, malgré des fonctions de scores différentes, sont globalement similaires, ils retournent la même superposition (4.3,4.1). Cet exemple illustre la convergence de résultats entre les différentes méthodes. Le TM-score de l'instance est très bas car celui-ci est basé sur la distance des paires alignées après alignement, hors comme la superposition est rigide, certaines paires alignées sont très éloignées.

TABLE 4.1 – Résultats de la comparaison des protéines 4cln(A) et 2bbm(A) avec quatre outils d'alignements structuraux (Apurva(CMO), PAUL, DALIX et MATRASX), via le serveur de comparaison CSA [116]
Les scores sont calculés à partir de la superposition issue de l'alignement des outils.
* dénote un score optimal

| Scores / Outils | CMO | PAUL | DALIX | MATRASX |
|---|---|---|---|---|
| Gap native score [%] | 0 | 0 | 73.4 | 128.497 |
| CMO score | 382* | 382 | 382 | 382 |
| PAUL score | 1161.690 | 1161.690* | 1161.690 | 1161.690 |
| DALI score | 481.231 | 481.231 | 481.231 | 481.231 |
| MATRAS score | 43344.957 | 43344.957 | 43344.957 | 43344.957 |
| CMO norm. similarity | [0.902,0.902]* | 0.902 | 0.902 | 0.902 |
| PAUL norm. similarity | 0.528 | [0.528,0.528]* | 0.528 | 0.528 |
| DALI norm. similarity | 0.339 | 0.339 | [0.339,0.589] | 0.339 |
| MATRAS norm. similarity | 0.212 | 0.212 | 0.212 | [0.212,0.485] |
| DALI z-score | 6.573 | 6.573 | 6.573 | 6.573 |
| TM-score | 0.161 | 0.161 | 0.161 | 0.161 |
| Aligned residues [#] | 148 | 148 | 148 | 148 |
| Aligned residues [%] | 100.000 | 100.000 | 100.000 | 100.000 |
| Coordinate RMSD | 14.781 | 14.781 | 14.781 | 14.781 |
| Distance RMSD | 10.838 | 10.838 | 10.838 | 10.838 |
| RMSD100 | 12.358 | 12.358 | 12.358 | 12.358 |
| Sequence identity [%] | 100.000 | 100.000 | 100.000 | 100.000 |

## 4.3 Alignements séquentiels basés sur la minimisation des distances inter-atomiques après superposition

Les approches de comparaison de structures les plus courantes se basent sur la superposition d'objets rigides ou l'assemblage de fragments superposés. A ces nombreuses méthodes telles que MAMMOTH [93], LGA [124] ou encore TMalign [126], Deepalign [113], CE [107] sont associées des fonctions de scores et il existe pratiquement une fonction de score par méthode. Cela entrave la comparaison de ces méthodes qui divergent dans leurs résultats et ne permettent pas de conclure quant au meilleur alignement. TMalign est l'un des outils les plus utilisés et par conséquent les plus critiqués dans la littérature. Cette heuristique sert de référence dans nombre d'études. Nous l'avons donc choisi comme outil de référence pour les outils d'alignements structuraux séquentiels (ordre dépendant).

### 4.3.1 TMalign, fonctionnement

TMalign [126] est une heuristique combinant superposition via matrice de rotation basée sur le TM-score [125] et programmation dynamique. La méthode se base sur les carbones alpha pour représenter les résidus de la protéine et débute par l'alignement des structures

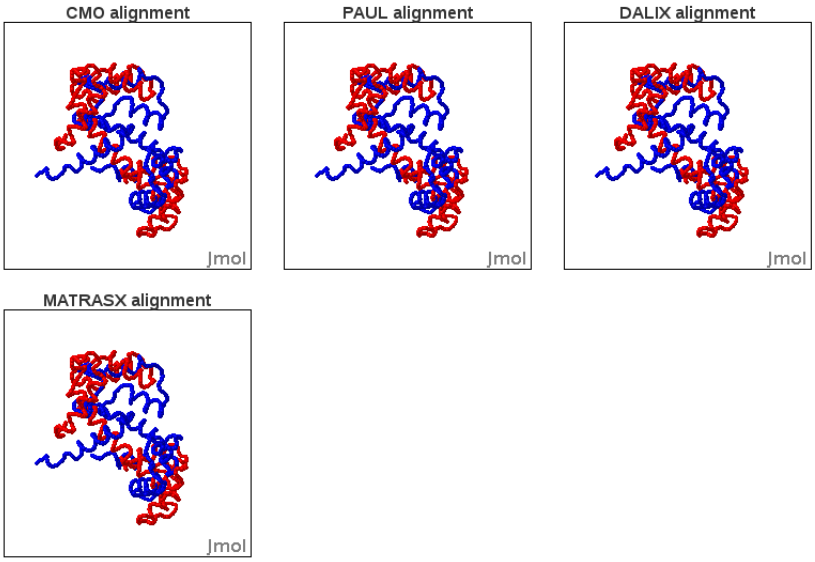



FIGURE 4.3 – Superspositions de 4cln(A) et 2bbm(A) basées sur les alignements obtenus avec Apurva (CMO), PAUL, DALIX et MATRASX

secondaires par programmation dynamique et par alignement de structures sans gaps. S'en suit une heuristique qui superpose les alignements initiaux puis cherche à maximiser le TM-score qui correspond à un RMSDc relatif à la longueur de l'une des protéines de l'instance (respectivement à l'autre protéine et propose également un TM-score moyen). Le TM-score a pour avantage cette relativité qui le distingue du RMSD. En effet le RMSD n'a de valeur que rapporté à la longueur de l'alignement obtenu. Fr-TMalign [94] est une variante de TMalign. Il diffère de TMalign au niveau de l'étape d'énumération des alignements initiaux qui sont ici des assemblages de bons fragments (c'est-à-dire ayant un score supérieur à un seuil). TMalign et ses variantes produisent des alignements qui ne sont pas soumis à un seuil de distances entre paires de résidus alignés, ainsi TMalign peut renvoyer un RMSDc élevé, néanmoins cette déviation est pondérée par la recherche du TMscore maximal qui élague les paires de résidus trop lointaines. Par conséquent, là où d'autres outils comme Apurva, incluent dans leurs résultats de larges mouvements de structure, TMalign restera plus local et produira un alignement plus court mais associé à un meilleur TMscore et un meilleur RMSDc. Si l'on reprend l'exemple de la section précédente, les quatre outils retournent un TMscore très faible (0.161) ce qui selon la littérature signifie que les deux structures n'ont pas la même topologie [118] alors que pour la même instance, TMalign renvoie un TMscore plus élevé (0.432) qui reste tout de même assez faible. De même, Tmalign trouve un alignement plus petit (72 paires de résidus contre 148 pour les autres) mais le RMSDc associé est plus faible (1.854 contre 10.838). TMalign est donc un outil qui se base sur une superposition rigide des protéines et détecte des sous-structures locales communes entre deux protéines.

### 4.3.2 Discussion

Ces outils d'alignements structuraux ont été évalués par la littérature et il en ressort qu'aucun d'eux ne fait réellement l'unanimité. Havrilla et Saçan [44] (2012) ont évalué entre autres TMalign, CE [107], Fatcat [120] et Smolign [44] (les deux derniers possédant une fonctionnalité supplémentaire décrite et discutée ci-dessous) et leur étude tend à prouver que TMalign retourne de meilleurs alignements structuraux en accord avec le TMscore ainsi que deux autres scores, le RMSDc et le Score SAS (tous les scores sont présentés en section 2.3). Ces approches séquentielles, rigides, ont un gros défaut qui est qu'elles bloquent sur trois cas : les mouvements importants de structures (charnières), les permutations de séquences et la répétition d'éléments au sein des structures. Le premier cas est dû au maintient de la déviation sous un seuil "tolérable", le deuxième est une conséquence directe de la contrainte d'ordre imposée aux algorithmes qui interdit les croisements dans l'alignement et le troisième est théoriquement détecté au cours de l'analyse mais les heuristiques ne renvoyant qu'un seul résultat elles écartent ainsi tous les alignements alternatifs.

## 4.4 Alignements non-séquentiels

Les alignements non-séquentiels, ou ordre-indépendants, sont apparus pour résoudre les cas où, au sein d'une instance, l'une des structures avait subi une permutation. Lors d'un événement évolutif, un morceau de séquence a été déplacé dans le gène codant pour la protéine, ce changement entre en conflit avec les outils d'alignements précédents qui ne prennent pas en compte ce cas et retournent donc des résultats incomplets. Abyzov et Ilyin [2] ont estimé qu'il y avait entre 17.4 et 35.2 % de permutations au sein des protéines ce qui justifie le besoin d'algorithmes performants pour détecter ces cas. Il existe plusieurs outils d'alignements non-séquentiels que nous séparons en deux catégories : rigides et flexibles.

- Les outils d'alignements rigides tels que MICAN [80], Gangsta [42], SANA [112] ou encore CECP [17], une version modifiée de CE [107] et DEDAL [33].

La détection de ces cas permet de trouver des homologues distants dont les séquences ont divergé mais dont le repliement a été maintenu. Parmi ces outils nous avons choisi d'utiliser MICAN comme point de comparaison afin de représenter les cas de détections de permutations circulaires.

### 4.4.1 MICAN

MICAN (Multiple-chains, Inverse alignments, $C\alpha$ only models, Alternative alignments, and Non-sequentials alignements) est un outil assez complet qui détecte de nombreux cas peu communs dans les structures comme les permutations L'objectif de l'algorithme décrit dans [80] est de trouver la matrice de rotation qui maximise les équivalences entre les structures comparées. MICAN est donc par nature un outil d'alignement rigide divisé en deux grandes étapes : l'alignement des structures secondaires et l'alignement des résidus.

L'alignement des structures secondaires (l'utilisation des structures secondaires) est courant dans une comparaison de structures car cela permet de réduire le nombre d'appariements

possibles entre les structures et accélère ainsi les algorithmes. Cet alignement se fait en quatre étapes.

- Calcul de l'appartenance de chaque résidu à une structure secondaire selon la méthode de Zhang et Skolnick [126].
- Calcul de tous les segments de structures secondaires (ensemble des fragments composés de 3 résidus appartenant à des feuillets $\beta$ ou bien de 5 résidus issus d'hélices $\alpha$) avec pour chaque segment quatre informations : le type ($\alpha/\beta$), les coordonnées du centroid et les vecteurs directionnels et perpendiculaires au vecteur directionnel.
- Les deux vecteurs calculés permettent de placer chaque segment dans un système de référence dont l'origine est localisée par les coordonnées du segment. Cette étape sert pour le hachage géométrique (reconnaissance sous-linéaire en nombre de modèles) qui suit.
- Cette dernière étape est un hachage géométrique qui sert à calculer les 50 meilleures matrices de rotations basées sur l'alignement de paires de fragments.

A la fin de l'alignement des structures secondaires, l'algorithme conserve 50 matrices de rotations candidates.

La seconde grande étape est l'alignement des résidus qui est elle-même constituée de deux étapes. La première est l'assignation de résidus équivalents après superposition des structures selon l'une des 50 matrices sélectionnées. Cela se fait en ne considérant que les paires de résidus (alignés) dont la distance est inférieure à un seuil donné (notée $\tau$ dans tout ce mémoire), puis MICAN construit une matrice de similarité attribuant un score dérivé du TMscore à chaque appariement de résidus potentiel avant de créer l'alignement à partir de celle-ci et de calculer un score associé dérivé du TMscore (mTMscore). La seconde étape consiste à affiner l'alignement en resuperposant les structures selon le dit alignement puis en recalculant un nouvel alignement à partir de la nouvelle superposition et ainsi de suite jusqu'à convergence du mTMscore.

Pour finir, l'algorithme retourne les 4 meilleurs alignements (meilleur signifiant mTMscore le plus élevé).

Afin de tester MICAN, Minami et ses collègues ont créé plusieurs jeux de données artificiels à partir de comparaisons séquentielles dont l'alignement de référence est connu ce qui permet de tester la qualité de l'outil. Nous avons utilisé l'un des jeux de données, MALIDUP-ns, en base de référence pour évaluer la qualité de différents outils d'alignement face à des cas de permutations. Nous avons entre autres enrichi les résutats de l'article en y ajoutant d'autres méthodes d'alignement non séquentiels comme SANA et CECP, ainsi que Flexsnap, un outil d'alignement flexible présenté ci-dessous.

## 4.5 Alignements flexibles, séquentiels et non-séquentiels

Les outils de construction d'alignements flexibles tendent à détecter et intégrer les charnières dans les structures qu'ils comparent. Cela car de nombreuses protéines ont une conformation variable selon leur environnement (la présence d'un ligand par exemple) [117]. La majorité des outils de détection des mouvements au sein des protéines sont séquentiels, comme FATCAT [119]. La plupart des outils sont basés sur une méthode de chaînage d'alignements

comme FlexProt [106] ou HingeProt [35]. Ces alignements rigides sont ensuite assemblés en alignements globaux. L'intérêt principal de ces algorithmes est qu'ils permettent non seulement de détecter une flexibilité d'une structure à l'autre mais aussi de passer outre cette flexibilité et de conclure que ces structures sont très proches. En effet, un outil rigide renverra ici une similarité moyenne alors que l'outil flexible montrera une forte similarité moyennant une ou plusieurs charnières. Nous avons choisi FlexSnap comme outil d'alignement flexible de comparaison. C'est un algorithme de chaînage d'alignements locaux non-séquentiel, par conséquent il devrait être en mesure de détecter les flexibilités mais aussi les cas de permutations circulaires et les cas linéaires "classiques".

### 4.5.1 FlexSnap

FlexSnap [102] est conçu pour créer un alignement global à partir de petits alignements locaux robustes. Ces petits alignements sont appelés AFP : well-Aligned Fragments Pairs. Les protéines sont découpées en fragments puis les fragments sont appareillés si possible d'une structure à l'autre pour former les AFPs. Les critères pour qu'un fragment d'une protéine A soit appareillé avec un fragment d'une protéine B sont :

- les fragments $F_A$, $F_B$, issus de A et B sont de la même longueur.
- le RMSDc correspondant à la superposition optimale des fragments est inférieur à un seuil $\epsilon$.

La seconde étape de l'algorithme est la création de l'alignement final en sélectionnant des AFPs. L'alignement se compose d'AFPs non chevauchants qui minimisent le nombre de gaps et de charnières tout en étant le plus long possible.

La force de FlexSnap est que les AFPs permettent d'avoir une base, un ensemble de similarités locales robustes qui en elles-mêmes fournissent déjà des informations utiles quant à l'existence de sous-structures communes entre deux protéines. Ensuite les assemblages permettent une vision plus globale. Théoriquement FlexSnap peut traiter tous les cas de comparaison de structures : l'alignement séquentiel étant un cas particulier de l'alignement non séquentiel. De plus l'alignement rigide est un sous-cas de l'alignement flexible, il s'agit d'un alignement dont la flexibilité est très faible. Par conséquent nous utiliserons FlexSnap comme outil de référence pour comparer et évaluer les résultats de nos outils.

## 4.6 Alignement de surfaces protéiques

Probis [62] détecte des sites similaires à la surface des protéines en effectuant des alignements locaux de structures (deux versions : protéine vs base ou 2 protéines). La comparaison s'effectue au niveau subrésiduel (groupes fonctionnels de Schmitt), les sites similaires correspondant à des motifs de propriétés physico-chimiques à la surface des protéines. La surface de la protéine est représentée par ses groupes fonctionnels. La comparaison de deux ensembles de groupes fonctionnels se modélise par un graphe produit, et est résolue en utilisant un algorithm de recherche de cliques maximum. Il génère ainsi des alignements locaux de structures et fournit un score pour chacun. L'algorithme de Probis tel que décrit dans Konc, Janezic 2010 est relatif à la comparaison d'une protéine face à la base de données, nous en présentons ici une version simplifiée, restreinte à la comparaison de deux protéines. L'algorithme contient quatre étapes :

i modélisation des surfaces des protéines par des graphes et sous graphes

ii création des graphes produit

iii recherche de la clique maximum pour chaque graphe produit

iv calcul des scores d'alignements obtenus.

La modélisation des protéines débute par (i) l'identification des résidus de la surface, puis, ces résidus sont représentés par des sommets selon les groupes fonctionnels décrits par Schmitt [105] et une arête est ajoutée entre deux sommets si leur distance est inférieure à 15 Å. Cette première valeur impose une recherche locale car ne connecte pas tous les résidus représentés en gros grain. Cinq types de sommets sont créés, correspondant aux cinq groupes fonctionnels : AC, accepteur d'hydrogène, DO, donneur d'hydrogène, ACDO, accepteur/donneur d'hydrogène, PI, aromatique et AL, aliphatique. Ces deux graphes sont ensuite divisés en n et m sous-graphes (correspondant respectivement au nombre de sommets dans chacun des graphes), chaque sous-graphe se définit par un sommet central et tous les sommets auxquels il est connecté (soit pour chaque groupe fonctionnel, les x groupes fonctionnels situés à une distance inférieure à 15 Å de ce groupe). Ensuite, pour chaque paire de sous-graphes dont les matrices de distances entre sommets sont assez similaires [62] le graphe de produit est créé (ii). Le graphe de produit correspond à l'appariement de sommets des sous-graphes de même type, et il existe une arête entre deux paires de sommets si les distances associées sont similaires à 2Å près. (iii) Une clique maximum de chaque graphe de produit est calculée, il s'agit du plus grand ensemble de sommets tous connectés les uns aux autres. (iv) Chaque clique maximum correspond à un alignement structural sur lequel il est possible de calculer plusieurs scores qui permettent d'élaguer les alignements insignifiants. Les surfaces sont recomposées par regroupements des alignements sélectionnés et les zones de similarités quantifiées.

Les expériences menées montrent la capacité de la méthode à détecter des sites de liaisons, qui sont des zones locales structurellement similaires, avec plus de précision que des algorithmes plus généraux comme Dalilite ou MolLoc [9]. La force de Probis se situe au niveau de la recherche hyper locale de similarités ce qui le rend totalement insensible aux

repliements globaux des protéines (les folds) et permet donc de détecter des similarités chez des protéines de structures éloignées.

## 4.7 Détection de répétitions structurales internes aux protéines

Le problème de détection des répétitions structurales est un peu différent du problème précédent (la comparaison de deux structures différentes) puisqu'ici c'est une étude locale de la même protéine. Ces répétitions peuvent s'observer au niveau de la séquence lorsque celle ci n'a pas trop évolué, d'où l'apparition d'outils de détection de séquences répétées tels que REPRO [36] ou IRIS [58] ou encore T-REKS [56] les répétitions en tandem. La fiabilité de ces méthodes est étroitement liée à la conservation primaire des motifs structuraux, plus la similarité de séquence entre motifs diminuera (soit plus les séquences divergeront), moins ces méthodes seront fiables. Or l'identité de séquence entre motifs peut être relativement faible (15%) ce qui a justifié le développement d'outils de détection de répétitions basés sur la structure tertiaire tels DAVROS [81], SWEIFE [1], ConSole [53], PRIGSA [25] ou encore l'outil de dallage (tessellation) de Parra *et al.* [95].

Nous avons principalement orienté nos études vers l'outil de Parra [95] car notre méthodologie est assez proche de la leur. En effet, ils commencent par rechercher des alignements locaux partiels à partir de fragments continus qu'ils nomment tuiles.

**Définition 4.1** *Une tuile est une séquence continue d'acides aminés.*

Toutes les tuiles possibles sont créées, de la tuile contenant la protéine entière aux tuiles de longueur 1. Chaque longueur de tuile correspond à un niveau indépendant, ainsi l'outil effectue les étapes suivantes pour toutes les tailles de tuiles. Ils utilisent ensuite ces tuiles comme requêtes dans l'outil TopMatch [109] qui retourne l'ensemble des sous-structures de la protéine qui s'alignent bien avec la tuile en les triant selon le score de TopMatch. Ces alignements permettent de trouver ensuite le plus grand ensemble de tuiles similaires recouvrant la protéine.

## 4.8 Discussion

L'alignement de structures protéiques de manière séquentielle est encore très utilisé car il concerne une majorité des comparaisons de structures mais nous connaissons aujourd'hui l'étendue de la variété des alignements. TMalign reste l'une, voire la référence, en termes de comparaison de structures. Cette heuristique rapide fournit de très bons résultats et est ainsi un outil de référence pour tester la puissance des nouveaux outils. Les cas où TMalign "se trompe" sont les cas de permutations circulaires et les cas de grandes flexibilités. Et même dans ces cas, nous ne pouvons pas vraiment parler d'erreur puisqu'il s'agit de cas que l'outil n'est pas conçu pour détecter. Les cas de permutations circulaires et de charnières ont été découverts bien plus tard que les cas "standards". Cela explique le faible nombre d'outils disponibles. Néanmoins les outils actuels montrent l'intérêt de la communauté pour ces cas et les différentes études ajoutent de plus en plus d'exemples à ces types de comparaisons.

Les cas de permutations circulaires notamment apparaissent de plus en plus. MICAN [80] est une heuristique assez puissante testée notamment sur les jeux de données MALIDUP et MALISAM, ce qui nous a poussés à la sélectionner pour nos comparaisons. Néanmoins nous avons fait une étude comparant les résultats de manière purement géométrique (longueur de l'alignement/RMSDc) sur MALIDUP-NS (non séquentiel) de différents outils (GANGSTA, CECP, MICAN, SANA, nos outils). Cette étude montre de meilleurs résultats pour SANA sur ce jeu de données mais cela pose la question de la pertinence de la question : est-ce qu'avoir le plus long alignement avec le plus faible RMSDc correspond au meilleur alignement ? Nous tenterons de répondre à cette question dans le chapitre dédié aux applications. Ce chapitre avait pour but de présenter une partie de l'état de l'art et plus particulièrement les outils d'alignements que nous avons utilisés pour comparer nos outils et trouver des pistes d'améliorations au sein des méthodes existantes. FlexSnap et l'outil de tiling (découpe en tuiles) montrent la pertinence de décomposer une comparaison de structures en petits alignements locaux robustes, TMalign et MICAN l'utilité des outils basés sur un alignement rigide de structures. La tendance qui se dégage est la modélisation de la flexibilité par la présence de point de charnière dans les alignements. Le relâchement de la contrainte de séquentialité est également un point important dans les nouveaux outils, que ce soit CECP (une modification de CE pour intégrer cet aspect non séquentiel), GANGSTA ou encore MICAN.

En conclusion nous avons des outils de comparaison qui nous permettent de détecter un grand nombre de cas différents d'alignements de structures et nous espérons ainsi avoir un aperçu global des capacités de nos outils.

## 4.9 Résumé du chapitre

Ce chapitre a permis d'observer l'état de l'art en matière d'outils de comparaisons de structures. Les premiers outils sont limités par la contrainte de linéarité de la séquence protéique, TMalign n'en reste pas moins l'un des outils les plus utilisés. Depuis, des outils de comparaisons comme MICAN détectent ces permutations de séquences et englobent donc les cas de permutations circulaires. En revanche MICAN est un outil rigide, par conséquent les cas de charnières ne sont pas détectés. Les algorithmes de chaînages de fragments comme FlexSnap détectent ces charnières et retournent des alignements non-séquentiels. Enfin il existe aussi des outils dédiés à la comparaison de structures face à elles-mêmes comme DAVROS. Nous avons une observation non exhaustive de l'état de l'art qui nous a permis de sélectionner trois outils : TMalign, MICAN et FlexSnap. Chacun symbolise une catégorie d'alignements (séquentiel, non-séquentiel, flexible) et ils vont nous permettre de comparer nos résultats aux leurs afin de comprendre et d'évaluer la pertinence des alignements structuraux renvoyés par nos outils.

# Chapitre 5

# Recherche d'éléments similaires par comparaison d'objets 3D modélisés dans un graphe

Ce chapitre est consacré à la présentation de Ninjas, un module de parcours de graphe qui recherche des **pseudocliques** à partir de **graines** (définition 5.8) au sein d'un **graphe d'alignement** (définition 5.6). Les premiers résultats liés à ce chapitre ont fait l'objet d'une publication dans [27].

## 5.1 Relation pseudoclique/alignement de points issus d'objets 3D

La notion d'alignement (cf définition 5.1), bien que présente dans le nom du graphe n'intervient pas dans ce chapitre. Si elle n'est pas essentielle pour les étapes de parcours du graphe par Ninjas mais elle est au cœur de cette dernière étape qui intervient après le parcours du graphe d'alignement, soit après la découverte des pseudocliques.

**Définition 5.1 (Alignement de deux ensembles)** *L'alignement de deux ensembles A et B :* $Al(A,B) = (p_i, p'_i), (p_2, p'_2), ..., (p_n, p'_n)$ *est un ensemble de couples de points (p,p') issus respectivement de A et B.*

D'après la définition d'une pseudoclique nous avons :
$pseudo(g) = v_i, v_2, ..., v_n$ et
$Al(A,B) = (p_i, p'_i), (p_2, p'_2), ..., (p_n, p'_n)$
Or on sait qu'à tout sommet $v \in pseudo(g)$ est associé deux points p et p' issus de A,B. **Donc, une pseudoclique de $G$ correspond à un alignement de A avec B. Par conséquent, Ninjas renvoie un ensemble d'alignements de A avec B.**

Deux types d'alignements sont possibles : alignements par appariements multiples (notés **alignements k à k** ) et alignements bijectifs, par défaut Ninjas retourne des alignements à

appariements multiples (dont la signification sera discutée ultérieurement) mais il est possible de filtrer les pseudocliques obtenues pour ne sélectionner que certains sommets et ainsi renvoyer un alignement bijectif ou **par paire** .

### 5.1.1 Alignement par appariements multiples ou alignement k à k

On associe à chaque pseudoclique $pseudo(g) \in G$ de taille *n* l'alignement correspondant $Al_{pseudo(g)} = (p_i, p'_i), ..., (p_n, p'_n)$. De même, sont par construction associés à $pseudo(g)$ deux sous-ensembles $A_{pseudo(g)} \subset A$ et $B_{pseudo(g)} \subset B$ contenant les points qui participent aux sommets de $pseudo(g)$. Tout point $p \in A_{pseudo(g)}$ (resp. $p' \in B_{pseudo(g)}$) participe à au moins un sommet de $pseudo(g)$. En revanche, une paire p,p' ne correspond qu'à un seul sommet $v_i \in pseudo(g)$. Cela induit que des points de $A_{pseudo(g)}$ et $B_{pseudo(g)}$ peuvent apparaître plusieurs fois dans l'alignement qui est alors nommé alignement k à k.

La définition 5.2 formalise cette propriété :

**Définition 5.2 (Alignement k à k)** *Soient deux sous-ensembles $A_{pseudo(g)}, B_{pseudo(g)}$. Un alignement k à k apparie à tout élément de $A_{pseudo(g)}$ (resp. $B_{pseudo(g)}$) au moins un élément de $B_{pseudo(g)}$ (resp. $A_{pseudo(g)}$).*

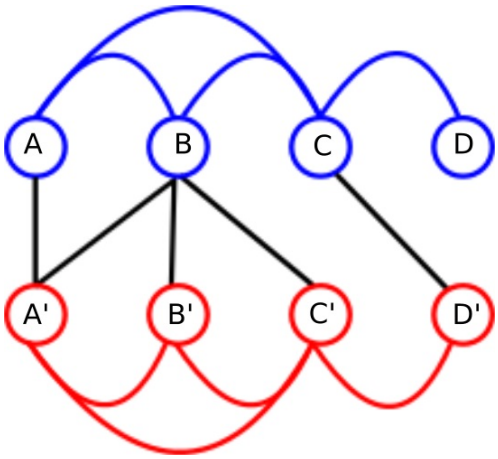



FIGURE 5.1 – Exemple d'alignement k to k, les sommets des pseudocliques (bleu et rouge) se lient pour certains avec plus d'un sommet. Le sommet $B$ est associé aux sommets $A', B', C'$.

### 5.1.2 Alignement bijectif ou alignement par paire

Contrairement à l'alignement k à k, un alignement par paire interdit la duplication des points (définition 5.3). Cela signifie que les éléments des sous-ensembles $A_{pseudo(g)} \subset A$ et $B_{pseudo(g)} \subset B$ participent à un seul et unique sommet de $pseudo(g)$.

**Définition 5.3 (Alignement par paire)** *Soient deux sous-ensembles $A_{pseudo(g)}, B_{pseudo(g)}$. Un alignement par paire couple tout élément de $A_{pseudo(g)}$ (resp. $B_{pseudo(g)}$) à un unique élément de $B_{pseudo(g)}$ (resp. $A_{pseudo(g)}$).*

### 5.1.3 Création de l'alignement bijectif à partir de l'alignement par appariements multiples

L'alignement à appariements multiples correspond à la pseudoclique obtenue après l'étape de filtrage des graines étendues. Cet alignement par défaut trouve sa justification dans des études de surfaces d'objets 3D (cf discussion générale), néanmoins, dans plusieurs cas à l'image de ceux présentés dans les chapitres suivants, un alignement par paire sera privilégié.

Soit $F(A_{pseudo(g)}, B_{pseudo(g)})$ la fonction d'appariement qui, à tout élément de $A_{pseudo(g)}$, associe un ou plusieurs éléments de $B_{pseudo(g)}$. Une fonction bijective sur le domaine de $F$ maximise le nombre de couples uniques entre $A_{pseudo(g)}, B_{pseudo(g)}$. La fonction bijective implémentée pour résoudre ce problème au sein de Ninjasest une implémentation de l'algorithme hongrois.

## 5.2 Définition principale de Ninjas

Soient A,B deux ensembles de points dans un espace euclidien à trois dimensions , A, B représentant deux objets 3D du même nom et soit G=(V,E) le graphe d'alignement de A avec B. Soit $pseudo(g)$ une pseudoclique de $G$ et soit $Al_{pseudo(g)} = (p_1, p'_1), (p_2, p'_2), ..., (p_n, p'_n)$ l'alignement correspondant.
Alors on a :

$$\begin{aligned} RMSDc(Al_{pseudo(g)}) &\leq \tau \\ RMSDd(Al_{pseudo(g)}) &\leq max(2\tau, \lambda) \\ |v, w| &\leq 2\zeta \ \forall v, w \in pseudo(g) \end{aligned}$$

## 5.3 Graphe d'alignement de deux objets 3D

Soient A et B deux ensembles de points représentant deux objets 3D dans un espace euclidien tri-dimensionnel. Leur analogie sera effectuée dans ce mémoire à travers un graphe d'alignement G=(V,E) avec $V$ l'ensemble des sommets et $E$ l'ensemble des arêtes de $G$ (définition 5.6), dont la construction sera décrite formellement dans le chapitre suivant .

### 5.3.1 Sommets du graphe d'alignement

Chaque sommet $v_{p,p'} \in V$ correspond à l'appariement d'un point p de A avec un point p' de l'ensemble B. La création du sommet $v_i$, dépend de la propension des points $p \in A$ et $p' \in B$ à se coupler et nous la définissons ici par la **Existence d'un sommet** (définition 5.4). $V$ représente donc l'ensemble des appariements compatibles entre paires de points issus respectivement de A et de B.

**Définition 5.4 (Existence d'un sommet, ou compatibilité d'une paire de points)** *Un sommet $v_i$ existe si les points correspondants peuvent s'appareiller.*

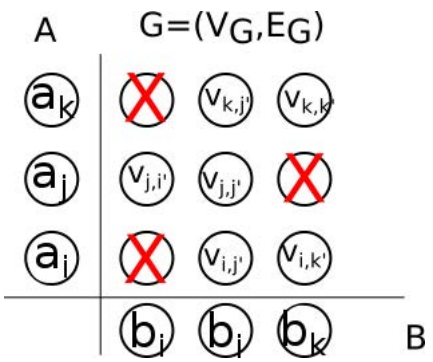



FIGURE 5.2 – Sommets du graphe d'alignement G=(V,E), ensemble des appariements possibles entre deux ensembles de points A et B.
$V = \{v_{i,j}, v_{i,k}, v_{j,j}, v_{k,i}, v_{k,k}\}$
Les sommets indiqués par une croix ne sont pas compatibles ; à l'inverse : un sommet indiqué $a_j, b_j$ est compatible

Cette définition caractérise le graphe qui ne contient donc que les sommets contenant des paires de points compatibles, ce qu'illustre la figure 5.2. Remarque : L'existence d'un sommet est relative à la nature des points considérés et traitée en amont de ce module. Nous négligeons donc dans ce chapitre les critères qui ont autorisé la création des sommets et utilisons ces sommets tels quels. La définition et l'emploi des critères sont décrits dans le chapitre consacré à Shinobi (chapitre 6).

### 5.3.2 Arêtes du graphe d'alignement

Une arête $e \in E$ existe entre deux sommets $v, w \in V$ si et seulement si les sommets respectent des critères spécifiques. La **création d'arête** (cf définition 5.5), elle correspond à la possibilité de créer ou non une arête dans le graphe selon la paire de sommets considérée.

**Définition 5.5 (Création d'arête ou compatibilité d'une paire de sommets)** *Une arête $e_{v,w}$ existe si et seulement si les sommets correspondants $(v,w)$ peuvent se lier.*

A l'instar des sommets, le graphe ne contient que les arêtes dont les sommets sont compatibles (cf figure 5.3). Les critères de compatibilité sont basés sur les distances entre les points des paires de sommets. Ces distances sont considérées de manière absolue (par rapport à une valeur seuil), ou relative par comparaison des distances entre paires de points. Chaque arête du graphe présente par construction deux propriétés relatives à des mesures de distance (mesurée ici en angströms) décrites ci-dessous.

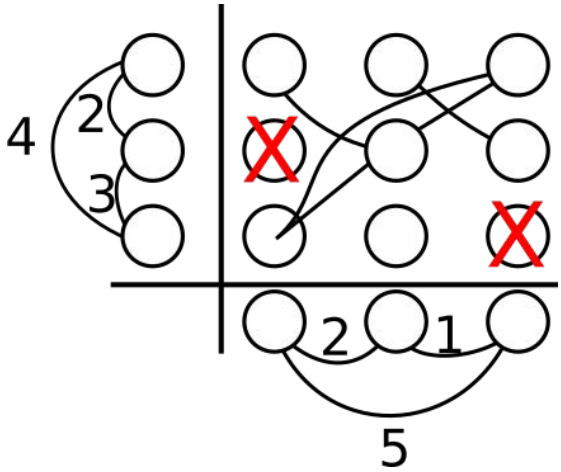



FIGURE 5.3 – Construction des arêtes du graphe d'alignement G=(V,E)
une arête dans le graphe est fonction de la compatibilité entre paires de points dans les ensembles A et B. Ici la compatibilité est symbolisée par la présence d'arêtes entre les points. Une arête dans le graphe existe si les paires de points correspondantes sont reliées par des arêtes. Ici, le critère de création d'une arête est la différence de distances (placées sur les arêtes des graphes de structures en abscisse et ordonnées). Si la différence entre les distances est supérieure à $\lambda = 1$ Å alors les paires de sommets ne sont pas compatibles et l'arête n'existe donc pas.

Soient quatre points $v_i(x_i, y_i, z_i), v_{i'}(x_{i'}, y_{i'}, z_{i'})$ deux points issus de l'ensemble A et $w_j(x_j, y_j, z_j)$ et $w_{j'}(x_{j'}, y_{j'}, z_{j'})$ deux points issus de l'ensemble B. Et soient deux sommets $v_{a,b}$ et $w_{1',b'} \in V$ du graphe d'alignement $G$. L'arête $e_{v,w} \in E$ possède les propriétés suivantes :

- la distance entre les points issus de A, $v_i$ et $w_j$ (resp. B, $v'_i$ et $w'_j$) est inférieure à une valeur seuil $\zeta$ (équation 5.1).

$$dist(v_i, w_i) \leq \zeta \ ; \ dist(v_{i'}, w_{j'}) \leq \zeta \tag{5.1}$$

- les distances entre paires de points issus du même ensemble sont identiques avec une erreur $\lambda$ (équation 5.2)

$$|dist(v_i, w_j) - dist(v_{i'}, w_{j'}| \leq \lambda \tag{5.2}$$

En résumé, une arête entre deux sommets de $G$ signifie (entre autres) que les paires de points issues de A et B considérées ont une distance identique plus ou moins $\lambda$Å et que leurs distances n'excèdent pas $\zeta$Å. Par conséquent ce que l'on nomme la compatibilité d'arête, c'est-à-dire la compatibilité d'une paire de sommets correspond à la compatibilité de paires de couples de points.

Remarque : Tout comme les critères de compatibilité de sommet, les valeurs $\zeta$ et $\lambda$ sont choisies lors de la création du graphe et ne sont donc pas discutées ici.

### 5.3.3 Définition du graphe d'alignement

Les définitions précédentes (5.4, 5.5) permettent de définir le graphe d'alignement de manière générique comme suit.

**Définition 5.6 (Graphe d'alignement.)** *Le graphe d'alignement de deux ensembles de points $G = (V, E)$ est un graphe non-orienté dont les sommets $v \in V$ modélisent l'ensemble des compatibilités entre paires de points issus de A et B respectivement et dont une arête $e \in E$ entre deux sommets $v, w \in V$ existe si les sommets sont compatibles.*

## 5.4 Graphe implicite du graphe d'alignement ou graphe de graines

Connaissant les propriétés du graphe $G(V, E)$, notamment celles induites par la présence d'une arête, on peut considérer un sous-graphe implicite $H = (V_H, E_H) \subset G(V, E)$. $H$ se construit en sélectionnant certains sommets et arêtes de $G$ répondant à des critères de distance plus restreints que ceux employés lors de la création des arêtes de $G$.

Soient $a, a' \in A$ et $b, b' \in B$ quatre points issus respectivement des ensembles A et B, $v_{a,b}, w_{a',b'} \in V$ sont les sommets de $G$ correspondants et $e_{v,w} \in E$. L'existence de $e_{v,w}$ assure que les propriétés des équations 5.1 et 5.2 sont respectées.

A présent soient $\zeta_s, \lambda_s, \zeta$ et $\lambda$ quatre valeurs seuils telles que $\zeta_s \leq \zeta$ et $\lambda_s \leq \lambda$ (discussion des rapports entre ces valeurs dans le chapitre 6).

Une arête $e_{v,w} \in E$ appartient à $E_H$ (et par extension les sommets $v, w \in V$ appartiennent à $V_H$) si et seulement si les distances correspondantes respectent les équations 5.1 et 5.2 dans lesquelles les valeurs $\zeta$ et $\lambda$ sont remplacées par $\zeta_s$ et $\lambda_s$ respectivement soit :

$$dist(v_i, w_j) \leq \zeta_s \ ; \ dist(v_{i'}, w_{j'}) \leq \zeta \tag{5.3}$$

$$|dist(v_i, w_j) - dist(v_{i'}, w_{j'}| \leq \lambda_s \tag{5.4}$$

Ces arêtes sont dites **restreintes** car elles *répondent à des critères de distance moins tolérants*. La définition du graphe implicite est donc la suivante :

**Définition 5.7 (Graphe implicite ou graphe de graines)** *Le graphe implicite $H = (V_H, E_H) \subset G = (V, E)$ d'un graphe d'alignement est un sous-graphe non-orienté constitué des arêtes $e \in E_H \subset E$ répondant aux critères de compatibilité d'arête restreinte (équations 5.3 et5.4). $V_H$ est l'ensemble des sommets $V$ induits par $E_H$.*

Ce graphe implicite se nomme le graphe de graines car Ninjas va parcourir ce graphe pour trouver de petits sous-ensembles qui vont être la base des pseudo-cliques. Ces ensembles se nomment formellement k-graines et sont définis par l'adjacence de tous les sommets de l'ensemble (définition 5.8).

**Définition 5.8 (Graine)** *Une graine de taille k (ou k-graine) du graphe implicite* $H = (V_H, E_H)$ *est un ensemble de k sommets tel que pour chaque paire de sommets* $v, w \in V_H$ *de l'ensemble il existe une arête* $e_{v,w} \in E_H$*. Une k-graine est donc une k-clique dans* $H$*.*

Remarque : Dans Ninjas, la taille choisie pour les k-graines est 3 sommets, ce choix est discuté dans l'article de Chapuis et collègues [26]. La taille de nos k-graines ne changeant pas, nous simplifions la notation en "graine".

## 5.5 Parcours du graphe d'alignement, recherche de pseudo-cliques avec Ninjas

Soient $G = (V, E)$ un graphe d'alignement et $H = (V_H, E_H) \subset G$ le graphe implicite de $G$. L'algorithme de Ninjas (algorithme 4) est constitué de trois étapes majeures : (i) l'énumération des graines du graphe implicite ($H$), (ii) l'extension de ces graines (création des pseudo-cliques) dans le graphe d'alignement ($G$) et enfin (iii) un filtrage des pseudo-cliques par sélection des sommets valides (sélection relative à la déviation tolérée). Ninjas renvoie entre une et n pseudo-cliques (n étant un paramètre défini par l'utilisateur) triées par taille décroissante.

**Algorithme 4** Ninjas Overview

```
function FIND_PSEUDOCLIQUES(G)
    Input : an undirected graph
    Output : A list of pseudocliques
    Result_list res_list = empty_result_list()
    Graph implicit_graph = create_implicit_graph(G)
    Seed_list seeds = enumerate_seeds(implicit_graph)
    for each seed in seeds do
        Vertex_set set = extend_seed(seed)
        Vertex_set result = empty_set()
        for each vertex in set  do
            if is_valid(vertex) then
                result.add(vertex)
            end if
        end for
        res_list.insert_if_better(set)
    end for
end function
```

Comme expliqué plus haut, dans Ninjas, une graine du graphe implicite $H = (V_H, E_H)$ est une 3-clique. Chacun des sommets correspondant à deux points a,b issus respectivement des objets A et B, une graine est donc un ensemble de six points appareillés deux à deux. Soit une graine $(v_i, v_j, v_k)$ composée de trois sommets et (i,i'), (j,j'), (k,k') les points correspondant respectivement aux sommets $v_i, v_j$ et $v_k$. Par construction, la condition de création

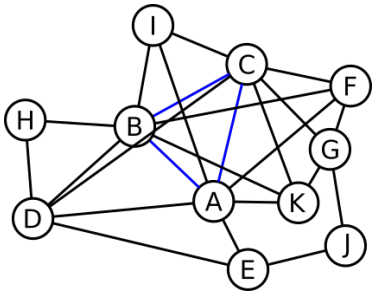



FIGURE 5.4 – Détection d'une graine (ABC, en bleu) dans un graphe non-orienté.

de la graine $(v_i, v_j, v_k)$ est l'existence des arêtes $e_{v_i,v_j} \in E_H, e_{v_i,v_k} \in E_H$, et $e_{v_j,v_k} \in E_H$. Si l'on reprend la signification ci-dessus d'une arête entre deux sommets du graphe $H$ (définition 5.7), une graine $(v_i, v_j, v_k)$ correspond à deux triangles (ijk) et (i'j'k') dont les distances d(ij),d(i'j'), d(ik),d(i'k') et d(jk),d(j'k') sont similaires deux à deux à $\lambda_s$Å près. Deux distances sont donc dites similaires si leur différence est inférieure à $\lambda$Å. Par conséquent, une graine $(v_i, v_j, v_k)$ est une 3-clique dans le graphe $H$ comme le montre la figure 5.4 et énumérer toutes les graines équivaut à rechercher toutes les 3-cliques de $H$.

Une première sélection des résultats a lieu lors de cette étape, les graines dont les triangles ont leur angle obtus trop proche de 180° (donc des points colinéaires ou presque colinéaires) sont élaguées par la méthode décrite dans l'algorithme 2 de [26]. Ce car par la suite, l'algorithme va chercher à superposer de manière optimale les triangles associés à ces graines et essayer de calculer la transformation (unique) permettant cette superposition. Or dans le cas de points colinéaires, il n'existe pas une unique transformation optimale mais une infinité de transformations.

A la fin de cette première étape, on obtient donc l'ensemble des graines du graphe $H$.

Remarque : la condition de distance inférieure à $\zeta_s$ entre points d'un même ensemble dans une graine contraint à la sélection de graines dites locales. Les points considérés sont proches dans l'espace. Cette proximité, ajoutée à la faible différence entre les distances d'un ensemble à l'autre crée des graines qui sont considérées robustes. Ce point est plus longuement abordé dans la discussion de ce chapitre.

L'extension des graines consiste, pour une 3-graine $g = (v_i, v_j, v_k)$ donnée, à trouver l'ensemble des sommets de $G$ connectés à $g$, soit l'union des k+1 cliques engendrées par la graine $g$ notées $cliques_{k+1}(g)$ . Pour cela, Ninjas liste les voisins (définition 5.9) de chaque sommet de la graine dans $G$ et fait l'intersection des trois ensembles obtenus. Cela équivaut à écrire :

$$pseudo(g) = voisins(v_i) \cap voisins(v_j) \cap voisins(v_k) \tag{5.5}$$

ou

$$pseudo(g) = \bigcup_{i=1}^{n} cliques_{k+1}(g) \tag{5.6}$$

**Définition 5.9 (Voisinage d'un sommet)** *Dans un graphe non-orienté G=(V,E), un sommet $w \in V$ est dit voisin d'un sommet $v \in V$ s'il existe une arête $e_{v,w} \in E$.*

Cette étape renvoie un ensemble de pseudo-cliques $pseudo(g)$ dont les sommets sont des candidats aux pseudos-cliques finales.

Les sommets des pseudocliques renvoyées ne sont pas tous conservés dans les pseudo-cliques finales, ce car les contraintes de sélection se font sur des mesures de distances internes à chaque ensemble A et B. Or deux points d'un sommet peuvent respectivement respecter les contraintes de distances posées par rapport à la graine mais ne pas se trouver du même côté du plan défini par les triangles des graines (voir figure 5.5). En conséquence les pseudocliques subissent ici une étape de filtrage pour identifier les sommets dans le cas cité, ces sommets sont dits non valides (définition 5.10) et retirés de la pseudoclique.

Soit $pseudo(g) = \{v_i, ..., v_n\}$ la pseudoclique de la graine $g \in H$ dans $G$ ; $v_i, ..., v_n \in V$ sont les sommets de $G$ connectés aux sommets de $g$. L'étape de filtrage débute par la recherche de la transformation optimale $T$ qui va, après superposition des triangles ijk et i'j'k' issus des sommets de $g$, minimiser les distances ii',jj' et kk'. Cette transformation est ensuite appliquée à l'ensemble des points $(p_i, p'_i)$ associés à chacun des sommets $v_i \in pseudo(g)$ de la pseudo-cliques et les distances $p_iT(p'_i)$ sont mesurées. Les sommets $v_i$ dont la distance $p_iT(p'_i)$ est supérieure à un seuil $\tau$ sont élagués. $\tau$ est une valeur (en Å) limitant l'éloignement des points couplés après superposition. Le bénéfice ici est double, d'une part il permet de supprimer les sommets $v_i$ composés de points $(p_i, p'_i)$ qui, après superposition, se trouvent chacun de chaque côté du plan défini par les triangles superposés de la graine (illustration sur la figure 5.5). D'autre part, cette sélection maintient la déviation globale moyenne sur les

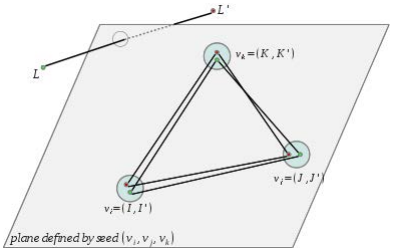



FIGURE 5.5 – [26]

coordonnées de la pseudo-clique (RMSDc) inférieure au seuil $\tau$ ce qui est un fort avantage dans le cadre des applications développées dans ce manuscrit.

**Définition 5.10 (Validité d'un sommet de la pseudoclique)** *Un sommet de la pseudoclique $pseudo(g)$ est dit valide si la distance entre ses points après la transformation $T$ est inférieure à un seuil $\tau$.*

L'ensemble des pseudocliques nettoyées correspond aux différentes zones de A et B géométriquement compatibles.

## 5.6 Complexité des étapes de Ninjas

Le problème initialement traité était le problème de recherche de cliques maximums, connu pour être NP-difficile [59]. Le relâchement de la contrainte de cliques en contrainte de pseudoclique a permis de réduire la complexité de certaines étapes majeures de Ninjas (cf [27]). Les graines étant filtrées en début de procédure, Ninjas est une heuristique . Ninjas étend l'ensemble des graines restantes et retourne toutes les pseudocliques obtenues triées par taille décroissante.

**Proposition 5.1** *Soit G=(V,E) un graphe non-orienté. L'étape d'énumération des graines (3-cliques) est au plus d'ordre* $O(|V|^3)$ *.*

*Démonstration* Enumération de 3-cliques dans un graphe non-orienté.
Soit $n = |V|$ le nombre de sommets du graphe G=(V,E).

$$\text{Nombre de 3-cliques dans un graphe} \frac{n!}{3!(n-3)!}$$

$$\begin{aligned}\frac{n!}{3!(n-3)!} &= \frac{n(n-1)(n-2)...(1)}{6(n-3)(n-4)(n-5)...(1)} \\ &= \frac{n(n-1)(n-2)}{6} \\ &= O(n^3)\end{aligned}$$

L'extension des graines est d'une complexité d'ordre O(|V|) puisque l'ajout d'un nouveau sommet à la pseudoclique a un coût de trois combinaisons. La dernière étape (le filtrage des sommets dans les graines étendues) est également d'ordre O(|V|) par graine étendue.

## 5.7 Propriétés géométriques des pseudocliques

Les pseudocliques issues de G=(V,E) ont trois propriétés :

- Le maintien de la déviation globale moyenne entre coordonnées après superposition sous un seuil $\tau$ (par construction) ; cette déviation est appelée Root Mean Square Deviation of coordinates ou **RMSDc**.

$$RMSDc \leq \tau \tag{5.7}$$

- Le maintien de la déviation globale moyenne entre distances internes (**RMSDd**) sous un seuil $max(2\tau, \lambda)$ ;

$$RMSDd \leq max(2\tau, \lambda) \tag{5.8}$$

- La garantie que les points issus des sommets de $pseudo(g)$ se trouvent à l'intérieur d'une sphère de diamètre $2\zeta$ ;

$$\forall i, j \in A_{pseudo(g)}, dist(i, j) \leq 2\zeta \text{ respectivement pour } \forall i', j' \in B_{pseudo(g)} \tag{5.9}$$

avec $A_{pseudo(g)} \subset A$ et $B_{pseudo(g)} \subset B$ sous-ensembles de A et B contenant les points participant aux sommets de $pseudo(g)$.

avec $\tau, \zeta$ et $\lambda$ des paramètres définis par l'utilisateur.

La première propriété est vraie par construction, les deux autres se démontrent en créant une abstraction de la pseudoclique $pseudo(g)$, un graphe représentant la clique associée à $pseudo(g)$. L'étude des arêtes manquant à la pseudoclique permet de déterminer ces propriétés.

### 5.7.1 Graphe enrichi associé à la pseudoclique

Soit $I_{pseudo(g)} = (V_I, E_I)$ le graphe associé à la pseudoclique $pseudo(g)$ issue du graphe G.

Par construction, $I_{pseudo(g)}$ contient tous les sommets et arêtes de $pseudo(g)$. Le graphe est complété par l'ajout de toute arête $e_{v,w}$ avec $v, w \in V_I$ deux sommets n'appartenant pas à la graine $g$, dite **arête i-spécifique**, de telle sorte que chaque sommet $v$ soit adjacent à tout sommet $w \in V_I$. Le graphe $I_{pseudo(g)}$ est donc la clique associée à la pseudoclique $pseudo(g) \in G$.

**Proposition 5.2** *Les arêtes i-spécifiques ont une propriété assurant que les distances entre points des sommets associés sont inférieures à la valeur seuil $2\zeta$ et une seconde propriété assurant que la différence de distance entre paires de points de même ensemble est inférieure à $2\tau$.*
*Soit $e_{v_{j,j'},w_{k,k'}}$ une arête i-spécifique.*
*Alors*

$$\begin{aligned} dist(j, j') &\leq 2\zeta \\ dist(k, k') &\leq 2\zeta \\ |dist(j, k) - dist(j', k')| &\leq 2\tau \end{aligned}$$

*Démonstration* Soient $v_{i,i'}, v_{j,j'}, v_{k,k'} \in V$ trois sommets avec (i,i'), (j,j'), (k,k') les points correspondants respectifs et $e_{v_{i,i'},v_{j,j'}}, e_{v_{i,i'},v_{k,k'}} \in pseudo(g)$ deux arêtes dans $pseudo(g)$. Et soit $e_{v_{j,j'},v_{k,k'}}$ l'arête spécifique complétant le triangle $v_{i,i'}, v_{j,j'}, v_{k,k'}$ ans $I_{pseudo(g)}$. Les arêtes issues de $pseudo(g)$ (et donc implicitement du graphe d'alignement $G$) ont une propriété basée sur la définition 5.1, qui assure que les distances entre points des sommets associés sont inférieures à la valeur seuil $\zeta$.

On cherche à connaître une borne sur les distances associées aux sommets $v_{j,j'}$ et $v_{k,k'}$ soient les distances $dist(j, k)$ et $dist(j', k')$. Par définition on a pour le triangle ijk (et respectivement pour le triangle i'j'k') :

$$\begin{aligned} dist(i, j) &\leq \zeta \\ dist(i, k) &\leq \zeta \end{aligned}$$

Ces distances étant des distances euclidiennes, on peut utiliser l'inégalité triangulaire pour trouver la valeurs seuil de la distance $dist(j,k)$ .

$$\begin{aligned}
&dist(j,k) \leq dist(i,j) + dist(i,k)\\
&dist(j,k) \leq \zeta + \zeta\\
&dist(j,k) \leq 2\zeta
\end{aligned}$$

La différence entre les distances $dist(j,k)$ et $dist(j',k')$ se calcule à partir des sommets $v_{j,j'} et v_{k,k'}$ après superposition. Soient donc quatre points $j, j', k, k'$ dans un espace euclidien tridimensionnel. Par construction (règle d'ajout d'un sommet à la pseudoclique $pseudo(g)$) on sait que :

$$\begin{aligned}
dist(j,j') &\leq \tau \qquad \text{et } dist(k,k') \leq \tau\\
dist(j,k) &\leq 2\zeta \qquad \text{et } dist(j',k') \leq 2\zeta
\end{aligned}$$

De plus la différence entre deux distances euclidienne est définie comme suit :

$$|dist(j,k) - dist(j',k')| = \begin{cases} dist(j,k) & - \quad dist(j',k') \\ dist(j',k') & - \quad dist(j,k) \end{cases} \tag{5.10}$$

or par inégalité triangulaire on a :

$$\begin{aligned}
&dist(j,j') \leq dist(j,k) + dist(k,k') + dist(j',k')\\
\Leftrightarrow\ &dist(j,j') - dist(k,k') \leq dist(j,k) + dist(j',k')\\
\Leftrightarrow\ &dist(j,j') - dist(k,k') \leq \tau + \tau\\
\Leftrightarrow\ &dist(j,j') - dist(k,k') \leq 2\tau
\end{aligned}$$

et réciproquement :

$$\begin{aligned}
&dist(k,k') \leq dist(j,k) + dist(j,j'') + dist(j',k')\\
\Leftrightarrow\ &dist(k,k') - dist(j,j') \leq dist(j,k) + dist(j',k')\\
\Leftrightarrow\ &dist(k,k') - dist(j,j') \leq \tau + \tau\\
\Leftrightarrow\ &dist(k,k') - dist(j,j') \leq 2\tau
\end{aligned}$$

On obtient donc le résultat suivant :

$$|dist(j,k) - dist(j',k')| = \begin{cases} dist(j,k) & - \quad dist(j',k') & \leq \quad 2\tau \\ dist(j',k') & - \quad dist(j,k) & \leq \quad 2\tau \end{cases} \tag{5.11}$$

Le graphe $I_{pseudo(g)}$ contient donc deux types d'arêtes, les arêtes $e_{v,w}$ issues directement de $pseudo(g)$ telles que la différence entre les distances , notée $|v-w|$ est inférieure ou égale au seuil $\lambda$ et les arêtes i-spécifiques où $|v-w| \leq 2\tau$. Par conséquent on assure qu'au sein de $I_{pseudo(g)}$, toutes les arêtes vérifient : $|v-w| \leq max(2\tau, \lambda) \forall v, w \in V_I$ Or, dans une clique d'un graphe d'alignement, si l'ensemble des arêtes de la clique vérifient la propriété sus-citée, alors la déviation globale moyenne basée sur les distances internes aux deux ensembles étudiés est inférieure à cette valeur seuil. Donc : $RMSDd \leq max(2\tau, \lambda)$.

## 5.8 Discussion

Ninjas recherche des pseudocliques au sein d'un graphe qui respectent certains critères et les énumère avant de retourner un ensemble d'alignements associé à ces pseudocliques. Les critères fournis sont purement géométriques et leur la signification est fonction du problème modélisé par le graphe d'alignement. Le premier point de discussion concerne le graphe implicite dont les critères de sélection plus restreints sélectionnent une partie de $G$ et limitent donc l'espace de recherche. Cette restriction est basée sur l'hypothèse que toutes les graines ne sont pas robustes, une graine dont les sommets sont à trois extrémités différentes des objets 3D étudiés nécessite une certaine tolérance (contrôlée par la valeur seuil de différence de distances entre les points) car les distances internes sont grandes. Cette flexibilité induite se propage ensuite à la pseudoclique dans sa globalité et créé une analyse plus grossière des objets. La notion de graine robuste (via ses valeurs seuils $\zeta$ et $\lambda$) est décidée en amont de Ninjas, de même que le seuil de superposition $\tau$. Notre hypothèse de base (partiellement validée par nos résultats) est qu'une sélection de petites graines locales est plus robuste pour trouver des zones similaires (locales ou globales ) au sein des objets. Laisser une trop grande flexibilité dès la graine initiale crée des pseudocliques qui, lors du tri final (par taille d'alignement et déviation globale moyenne sur les coordonnées) seront mal classées et donc écartées. Les graines locales, avec une faible déviation initiale renverront des alignements également moins déviants.

Ninjas est une heuristique, il ne garantit pas de trouver le meilleur alignement de deux objets 3D, sa force réside dans le fait qu'il retourne plusieurs alignements pertinents (selon les critères donnés) des deux objets. La notion même de "meilleur alignement" de deux objets 3D est discutable, est-ce le plus long alignement sous une déviation globale choisie ? Est-ce l'alignement qui maximise sa taille en essayant de minimiser cette déviation ? Et dans ce cas où est la limite ? Dans le domaine applicatif principal de Ninjas, l'étude des protéines en trois dimensions, cette question est encore en suspens. La communauté n'a pas encore tranché et il est largement admis qu'actuellement aucune méthode de comparaison de protéines ne s'est imposée. Notre choix fut de chercher de bons alignements avec une déviation contrôlée pour deux raisons : la première étant qu'un alignement à biais contrôlé est interprétable, il signifie qu'une zone est similaire au sein de deux objets à $\tau$Åprès. La seconde est qu'au lieu de chercher une similarité globale qui peut s'avérer très flexible comme le montreront les exemples d'applications, il est très simple de chercher avec Ninjas plusieurs zones locales similaires et ensuite de les associer pour reconstruire les objets.

En conclusion, Ninjas est un solveur de pseudocliques puissant qui ne retourne non pas

une solution mais plusieurs et permet ainsi une étude plus exhaustive du graphe d'alignement fourni.

## 5.9 Résumé du chapitre

Nous avons présenté dans ce chapitre Ninjas, un outil de recherche de pseudocliques dans un graphe non orienté. Notre méthode se base sur des graines choisies selon des critères de robustesse ( bonne superposition des sommets de la graine) puis étendues. Ces graines étendues forment des pseudocliques car chaque sommet de la pseudoclique est assurément connecté à la graine mais pas obligatoirement aux autres sommets (cela n'est pas vérifié) ; par cela Ninjas est une heuristique. Ce sont ces pseudocliques qui correspondent dans notre cadre applicatif à des alignements structuraux. Ninjas permet de rechercher des alignements globaux ou locaux en maintenant des valeurs de RMSDc et RMSDd en dessous des seuils choisis par l'utilisateur. De plus notre outil retourne non pas un mais plusieurs alignements pour chaque comparaison. A partir de ces alignements structuraux, nous allons pouvoir analyser les similarités entre nos structures protéiques.

# Chapitre 6

# Modélisation de la comparaison structurale de protéines par un graphe d'alignement

La grande majorité des outils de comparaison de structures protéiques réduisent les macromolécules à un ensemble ordonné de points dans l'espace et oublient totalement ou presque les propriétés physico-chimiques intrinsèques des objets qu'ils considèrent. Cette réduction se justifie par le largement admis paradigme structure-fonction des protéines qui dit que la structure de la protéine porte sa fonction. Ainsi deux protéines avec des structures similaires vont porter des fonctions similaires. Quelques années consacrées à l'étude des protéines m'ont montré que la biologie moléculaire est un peu comme la grammaire française, il y a les règles et à chaque règle ses exceptions. Le paradigme structure-fonction n'y échappe pas. En effet, les nombreuses recherches ont décelé des protéines qui avaient des structures similaires et des fonctions différentes, notamment chez les enzymes, et à l'inverse des protéines de structures différentes effectuant des fonctions similaires (convergence évolutive). Suite à ces constats, nous avons cherché à intégrer les propriétés physico-chimiques des protéines dans nos outils de comparaison de structures. Dans ce chapitre nous présentons Shinobi, un module qui modélise la comparaison de deux protéines (de manière globale ou plus spécifique selon la problématique) au sein d'un graphe d'alignement.

Soient deux protéines A et B, le fonctionnement de Shinobi est le suivant : (i) création (virtuelle) des **graphes de structure** associés aux protéines, (ii) couplage des graphes en un graphe d'alignement (cf Algorithme 6).

De nombreux paramètres dépendant de la question biologique posée en amont de la comparaison entrent en compte, ce chapitre présente les différents modèles de représentation et paramètres puis les discute. Le chapitre dédié aux études de cas présente les différentes combinaisons suivant les applications et les résultats associés seront discutés. Notre but ici était de pouvoir injecter de l'information biologique dans un graphe. Cela passe par l'ajout d'étiquettes sur les différents sommets ainsi que par des appariements contraints. Les contraintes sont directement issues des propriétés biologiques modélisées par ces étiquettes.

Shinobi Overview

```
function PROTEIN_MODELISATION(ProteinStructure_1, ProteinStructure_2, Parameters user_parameters)
    return Graph_Structure graph_1, Graph_Structure graph_2
end function
function CREATE_ALIGNMENT_GRAPH(graph_1, graph_2, user_parameters)
    Alignment_Graph graph
    for a_vertex in graph_1 do
        for another_vertex in graph_2 do
            if vertex_compatibility(a_vertex, another_vertex, user_parameters) == True then
                Vertex v = (a_vertex, another_vertex)
                graph.add(v)
            end if
        end for
    end for
    for vertex in graph do
        for another_vertex in graph do
            if edge_compatibility(a_vertex, another_vertex, user_parameters, graph_1, graph_2) == True then
                Edge e = (a_vertex, another_vertex)
                graph.add(v)
            end if
        end for
    end for
    Output : An Alignment Graph
end function
```

## 6.1 Modéliser une protéine (3D) par un graphe de structure

Une approche simple pour modéliser une protéine est de la représenter par ce que l'on nomme ici un **graphe de structure** (cf Définition 6.1), c'est-à-dire un graphe représentant une protéine entière ou une portion de protéine par un ensemble de sommets et d'arêtes.

**Définition 6.1** *(Graphe de structure)*
*Le graphe de structure $S = (V_S, E_S)$ d'une protéine P est un graphe non-orienté dont les sommets $V_S$ symbolise un ou plusieurs atomes de P (nommés **unités structurales)** et l'arête $e_{v,w} \in E_S$ est pondérée par la distance euclidienne mesurée entre les deux unités structurales. Toutes les arêtes existent initialement, ensuite nous pouvons choisir de supprimer les arêtes représentant des distances inférieures ou supérieures à un seuil. Cela permet de concentrer par la suite la modélisation sur des similarités locales ou globales.*

Une protéine est de base un ensemble d'atomes interagissants comme expliqué dans le chapitre 1.

Le choix du ou des groupes d'atomes représentés par un sommet est crucial et dépend de la question biologique posée. Nous présentons ici les différents sous-modèles de graphes de structure basés sur ce choix des sommets ainsi que les attributs associés et les coordonnées définies.

### 6.1.1 Modèle résiduel ($C\alpha$ ou $C\beta$)

Le modèle résiduel (nommé modèle $C\alpha$dans cette thèse) est le plus connu et le plus utilisé dans le cadre de la comparaison de structures protéiques. La protéine est modélisée par l'enchaînement des carbones centraux ($C\alpha$) de ses acides aminés. A chaque sommet du graphe va correspondre un résidu qui hérite des coordonnées atomiques de son carbone central et des propriétés physico-chimiques de l'acide aminé considéré (hydrophobicité, polarité, ...). On ajoute également l'appartenance à une structure secondaire (ou non), cette étiquette est attribuée via l'utilisation de DSSP [57], un outil qui analyse la structure d'une protéine et détermine pour chaque acide aminé la structure secondaire à laquelle il appartient. Les arêtes possèdent un seul attribut, un poids correspondant à la distance entre les $C\alpha$considérés.

A l'instar du modèle $C\alpha$, le modèle $C\beta$est également disponible, il est identique au précédent à ceci près qu'ici les coordonnées attribuées à chaque sommet sont celles du $C\beta$et non du $C\alpha$. Dans le cas de la glycine qui ne possède pas de $C\beta$, nous avons fait le choix de conserver les coordonnées du $C\alpha$.

### 6.1.2 Modèle résiduel mixte ($C\alpha C\beta$)

Ce modèle mixte est très similaire au précédent, il possède un attribut supplémentaire notable au niveau des sommets qui est la conservation des coordonnées du $C\beta$. De même les arêtes ne sont plus seulement pondérées par la distance $C\alpha$- $C\alpha$mais également par les distances $C\alpha$- $C\beta$, $C\beta$- $C\alpha$et $C\beta$- $C\beta$.

Les résidus sont représentés par leurs $C\alpha$, néanmoins, pour qu'une arête existe entre deux sommets du graphe d'alignement, toutes les distances citées ci-dessus doivent respecter les critères de distances ($\lambda$, la différence de distances entre paires de sommets, $\zeta$, la distance absolue entre deux résidus). Une seule arête représente tous ces critères, si un seul n'est pas respecté, l'arête n'existe pas.

### 6.1.3 Modèle gros grain, ou modèle selon les groupes fonctionnels définis par Schmitt *et al.* [105] (FGS)

Les groupes fonctionnels sont une méthode de représentation des acides aminés à mi-chemin entre le $C\alpha$et le modèle atomique. Un acide aminé est séparé en plusieurs groupes de un ou plusieurs atomes lourds. Nommés ainsi pour les différencier des groupes fonctionnels définis en chimie, ils sont composés d'un ou plusieurs atomes lourds (et hydrogènes associés). Ils portent les propriétés des acides aminés et décrivent plus précisément l'occupation de l'espace par les acides aminés. Dans le cas d'un groupe fonctionnel poly-atomique, les atomes sont barycentrés au sein d'un seul représentant pour les besoins du modèle mais toutes les propriétés sont conservées implicitement. A la différence des modèles précédents, le modèle en groupes fonctionnels possède un nombre d'attributs restreint : la nature du groupe fonctionnel (A,DA,D,AL,PI) et l'appartenance à une structure secondaire (attribut discutable).

### 6.1.4 Modèle atomique

Le modèle atomique, comme son nom l'indique, associe à chaque atome lourd de la protéine un sommet. De base, le sommet hérite des attributs du résidu auquel l'atome appartient (incluant la structure secondaire) en plus des attributs physico-chimiques (charge, rayon de Van der Waals, etc) et spatiaux (coordonnées) propres à l'atome. Comme dans le modèle résiduel, l'arête liant deux sommets est pondérée par la distance entre les atomes correspondants.

Ce modèle atomique permet de représenter finement la protéine mais entraîne la création de graphes de structures de plusieurs milliers de sommets ce qui entraîne par la suite des points techniques de quantité de mémoire virtuelle non négligeables. De plus, une représentation trop fine peut négliger les interactions locales fonctionnelles. En effet une fonction chimique telle qu'un cycle aromatique résulte de l'interaction des atomes de carbone et d'azote le constituant, considérer ces atomes un à un équivaut soit à répéter l'information, soit à l'ignorer dans les annotations de sommets. En revanche, ce type de modélisation est assez précis au niveau de l'étude géométrique des structures car les distances inter-atomiques sont moins grossières que les distances entre barycentres de groupes d'atomes (modulo les flexibilités dues aux données brutes).

### 6.1.5 Pertinence des attributs ajoutés aux sommets du graphe de structure

Techniquement il est possible d'ajouter n'importe quelle propriété aux sommets d'un graphe. Les difficultés se situent au niveau de la pertinence des propriétés choisies et plus tard, lors du couplage de deux graphes, du poids et de l'importance à accorder à une propriété

par rapport à une autre. De nombreuses propriétés physico-chimiques ont été envisagées, des plus larges comme l'appartenance du sommet à une structure secondaire comme les plus fines à l'image du nombre d'atomes lourds correspondant au sommet. Elles n'ont pas pu être toutes testées mais méritent un certain intérêt, notamment dans de futures études.

La plupart des propriétés physico-chimiques des acides aminés ont été étudiées et ajoutées aux modèles résiduels et résiduels enrichis. Les principales propriétés sont l'appartenance à une structure secondaire, la polarité, l'hydrophobicité du résidu considéré, la charge ou encore le volume (minuscule/petit/gros) selon les classifications communément admises. L'un des soucis qui sera rediscuté dans la section graphe d'alignement est le poids à accorder à chaque propriété, nous pouvons considérer que l'appartenance à une structure secondaire est indépendante de l'hydrophobicité mais ce n'est pas le cas entre les différentes propriétés chimiques. Est-il plus important de conserver telle ou telle propriété ? Si oui laquelle ? Les expériences préliminaires menées n'ont pas permis de conclure de manière certaine, pour l'instant ma conclusion est "cela dépend des cas", ce qui n'est pas très simple à traduire de manière algorithmique. L'une des solutions envisagées est de passer par l'utilisation des matrices BLOSUM ou assimilées pour estimer la substituabilité d'un résidu par un autre au niveau des modèles $C\alpha$et $C\alpha C\beta$. Cette méthode n'a pas encore fait ses preuves mais trône fièrement dans les perspectives d'études. La représentation en groupes fonctionnels de Schmitt est restreinte à la nature des FGS et à l'appartenance d'un FGS à une structure secondaire (bien que cela soit discutable et discuté plus loin).

| | Propriétés | | | | |
|---|---|---|---|---|---|
| Modèle | SSE | Hydrophobicité | Polarité | Substituabilité | Nature du groupe fonctionnel |
| $C\alpha$ | x | x | x | x | |
| $C\alpha C\beta$ | x | x | x | x | |
| FGS | x | | | | x |

TABLE 6.1 – Propriétés des unités structurales

En résumé, pour l'instant, la plupart des propriétés physico-chimiques "classiques" des acides aminés sont implémentées dans les graphes de structure des modèles $C\alpha$et $C\alpha C\beta$, la nature des FGS est la propriété principale du modèle du même nom et tous les modèles acceptent l'ajout de la structure secondaire à laquelle appartiennent les éléments.

### 6.1.6 Création d'arêtes entre les sommets du graphe de structure

Par défaut, il existe une arête pour chaque les paires de sommets du graphe, celle-ci est pondérée par la distance euclidienne entre ses sommets. Il est néanmoins possible d'interdire ces arêtes grâce à des valeurs seuils éliminant les distances trop faibles ou trop grandes. Ces éliminations sont pertinentes lorsque l'on cherche à aligner localement les structures ; lors de la recherche de sites actifs ou de sites de liaison par exemple. Cette restriction a non seulement un but d'accélération des algorithmes (car réduit l'espace de recherche de solutions) mais permet également de filtrer en amont de futurs appariements géométriquement corrects mais biologiquement faux pour la question biologique considérée.

L'ordre dans lequel sont rangés les éléments au sein de la protéine et du graphe associé peut avoir de l'importance, notamment dans le cas de comparaisons séquentielles de structures. La règle employée ici est l'ordre d'apparition des unités structurales dans le fichier de structure protéique (PDB), (figure 6.1).

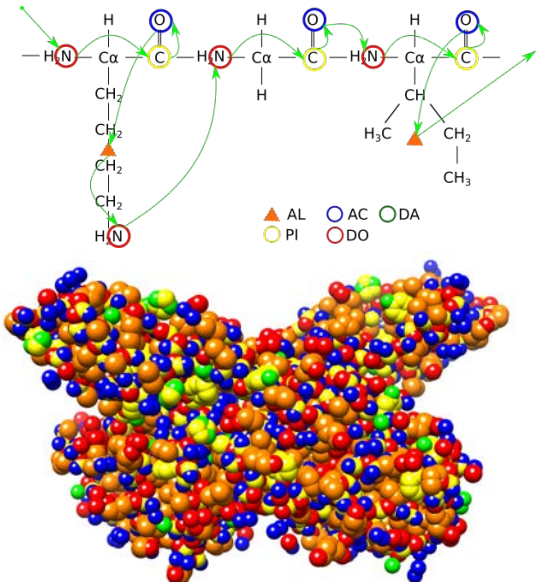



FIGURE 6.1 – Ordonnancement des unités structurales (ici les groupes fonctionnels), à gauche, et représentation d'une protéine (3BIO) par ses groupes fonctionnels (à droite)
AL : Groupe aliphatique (orange), PI : cycle aromatique (jaune), AC : accepteur d'hydrogène (bleu), DA, donneur-accepteur d'hydrogène (vert), DO : donneur d'hydrogène (rouge).
Le chemin en vert (figure du haut) modélise l'ordre dans lequel sont rangés les groupes fonctionnels.

## 6.2 Modéliser la comparaison de deux protéines : couplage de graphes de structure dans un graphe d'alignement

Comparer deux protéines correspond à coupler leurs graphes de structures correspondants en un graphe d'alignement qui va englober toutes les possibilités d'appariements et surtout élaguer celles qui ne correspondraient pas à la recherche initiale.

Soient $S_A = (V_{S_A}, E_{S_A})$ et $S_B = (V_{S_B}, E_{S_B})$ les graphes de structure des protéines A et B. $G(S_A, S_B) = (V_G, E_G)$ est le graphe d'alignement correspondant au couplage des deux graphes de structure et donc à la modélisation de la comparaison de A et B. Si nous reprenons la Définition 5.4, ajouter un sommet dans G équivaut à trouver une paire de sommets de $S_A$ et $S_B$ compatible. De même, pour créer une arête entre deux sommets de G, il faut que les arêtes entre les sommets des graphes de structure soient compatibles (définition de la

compatibilité d'arête : Définition 5.5-). Cette partie du manuscrit définit les compatibilités en fonction du modèle utilisé ainsi que leurs implications dans les futurs alignements.

### 6.2.1 Compatibilité d'arête, critères de distances

Soient deux sommets $v_{ij}$ et $w_{i'j'} \in V_G$ du graphe d'alignement modélisant les appariements des sommets $i, i' \in V_{S_A}$ avec les sommets $j, j'' \in V_{S_B}$ des graphes de structure $S_A, S_B$. L'existence d'une arête $e_{v_{ij},w_{i'j'}} \in E_G$ est soumise à deux critères qui ont déjà été formalisés par les équations 5.1 et 5.2.

- La distance $d(i, i')$ (resp. $d(j, j')$) est inférieure ou égale à $\zeta$Å(équation 5.1), cela contraint la recherche sous-structures similaires à une boule de diamètre $\zeta$ et permet des recherches locales de similarités.
- La différence entre les distances $d(i, i')$ et $d(j, j')$ est inférieure ou égale à $\lambda$ Å(équation 5.2).

### 6.2.2 Compatibilité de sommet, critère structuraux et physico-chimiques

Globalement, un sommet $v_{i,j} \in V_G$ existe dans le graphe d'alignement G=(V,E) si les unités structurales $i, j$ issues respectivement des graphes de structure $S_A$ et $S_B$ sont compatibles. La compatibilité résulte du passage du sommet candidat dans une série de critères de l'outil. Le choix de ces critères dépend du niveau de représentation auquel on se trouve ainsi des objectifs de l'analyse. Ainsi, Shinobi possède des règles d'appariement de sommet modélisant une complémentarité ou à l'inverse une similarité, des critères structuraux, des critères chimiques. Techniquement chaque critère peut être appliqué pour n'importe quel modèle et quelle que soit la question biologique posée, néanmoins certains critères conviennent mieux à certains modèles que d'autres.

### 6.2.3 Utilisation du module, couplage avec un solveur de graphe

Shinobi modélise une question biologique dans un graphe. C'est une étape essentielle dans la comparaison de protéines car c'est dans ce graphe que les zones d'intérêts sont répertoriées. Cependant l'obtention du graphe seul n'est pas suffisante pour répondre à la question biologique, il est nécessaire de parcourir le graphe pour en extraire les alignements. Shinobi a été conçu pour pouvoir être utilisé avec trois solveurs (et tous les solveurs similaires) :

- ACF [85] (Alignment Clique Finder) est un solveur de cliques de l'équipe Symbiose qui cherche le plus long alignement croissant (la plus grande clique) au sein de G=(V,E).
- Cliquer [84] [1], un solveur de cliques développé par Ostergaard, il possède comme atout l'énumération de toutes les cliques de taille spécifiée et le renvoi d'une ou plusieurs cliques maximales.
- Ninjas 5, est un solveur de pseudocliques de l'équipe GenScale. Il retourne des pseudocliques issues de G=(V,E), en garantissant certaines propriétés décrites dans le chapitre dédié.

1. Le format de graphe est le format dimacs standard, utilisable donc par d'autres solveurs

TABLE 6.2 – Critères de compatibilité de sommet du graphe d'alignement et domaines d'applications

| Critère | Modèles associés | Fonctionnement | Domaines d'application |
|---|---|---|---|
| Correspondance des structures secondaires | $C\alpha$ $C\alpha C\beta$ | 1. interdit l'appariement feuillet/hélice ou 2. oblige l'appariement $\mathcal{S}/\mathcal{S}$, $\mathcal{H}/\mathcal{H}$, $\mathcal{O}/\mathcal{O}$ | Recherche de similarités globales ou locales entre deux structures complètes (par opposition aux surfaces) |
| Similarité des groupes fonctionnels | FGS | Apparie les groupes fonctionnels partageant au moins une propriété | Recherche de similarité globale/locale de structures ou de surfaces |
| Complémentarité des groupes fonctionnels | FGS | Appariements PI/PI, AL/AL, D/A, D/DA, DA/DA, A/D (partage d'hydrogènes) | Docking (surfaces) |

ACF fut utilisé dans les études préliminaires puis remplacé par Ninjas. Les différences majeures entre Ninjas et Cliquer sont les suivantes :

- Le RMSDd des alignements correspondants aux résultats est garanti inférieur à un seuil : Cliquer, par la recherche de cliques assure que la différence entre paires d'unités structurales est inférieure à $\lambda$, Ninjas, par construction relâche cette contrainte à $2\lambda$. Ce seuil $\lambda$ correspond à la marge autorisée lors de la comparaison de deux distances (ce qui conduit à la création d'une arête).
- Les valeurs de RMSDc ne sont garanties que par Ninjas via son paramètre interne $\tau$, Cliquer ne possède pas de telle garantie car énumère des cliques dans un graphe et néglige le contenu des sommets/arêtes.
- Cliquer est un algorithme exact, parcourant par conséquent l'intégralité du graphe, Ninjas en revanche est une heuristique qui garantit de trouver de bons résultats sans assurer que ce soient les meilleurs.

La garantie du RMSDc nous a fait choisir Ninjas dans plusieurs cas d'étude comme la recherche de similarités globales ou locales de structures ou de surfaces, Cliquer trouvant son intérêt dans la recherche de répétitions structurales internes (8).

### Utilisation conjointe de ShinobiNinjas (ShiNi) : procédure

Pour exploiter et explorer le graphe d'alignement modélisant la comparaison de deux protéines, Ninjas requiert deux graphes.

- Le graphe G=(V,E) créé par Shinobi, avec des critères de compatibilités sur les sommets ainsi que les critères sur les distances absolues ($\zeta$) et relatives ($\lambda$).
- Un graphe, dit graphe de graines, qui correspond à des zones locales très similaires d'une structure à l'autre. La création de ce second graphe s'effectue avec Shinobi, les critères d'appariement des unités structurales (la création des sommets donc) ne varie pas mais le choix des arêtes est plus restreint tant au niveau de la distance absolue ($\zeta s$) entre unités structurales d'une même protéine qu'au niveau de la variation de distance d'une paire d'$US$ à l'autre ($\lambda s$). Comme expliqué dans le chapitre 5, ces restrictions ont deux objectifs : restreindre la recherche d'alignements aux plus pertinents en se basant sur une graine robuste (autrement dit peu flexible) et ainsi accélérer le logiciel.

Le premier graphe autorise une certaine flexibilité et c'est en son sein que l'on va rechercher les alignements correspondant aux sous-structures similaires entre les deux protéines. Le second rassemble les graines qui servent à déterminer les superpositions testées. Ninjas parcourt donc les graphes et retourne le nombre d'alignements souhaités tels que chaque alignement a un RMSDc inférieur ou égal à $\tau$ et un RMSDd sous le seuil $2\lambda$ et que toutes les unités structurales d'une même structure sont au plus éloignée de $\zeta$ Å

.

### 6.2.4 Application des modèles

Le couple ShinobiNinjas (ShiNi) est destiné à une comparaison approfondie de deux structures protéiques (pour une analyse plus générale se référer au chapitre 4). Les heuristiques de comparaison de structures ont fait leurs preuves à travers la littérature, traitant chacune un type de cas spécifique. ShiNi permet de pousser plus loin l'analyse en retournant des alignements alternatifs, des combinaisons d'alignements mais surtout s'éloigne du squelette des protéines pour une analyse davantage basée sur les propriétés physico-chimiques.

Le modèle atomique s'adresse à la comparaison de surfaces en règle générale, notamment à la recherche de sites de liaison. Shinobi ne détecte pas automatiquement la surface des protéines, il est nécessaire d'effectuer cette étape en amont. Nous utilisons vorlume [24], un outil développé par l'équipe ABS [2] pour sélectionner les atomes de la surface d'une protéine. Nous l'avons testé sur une instance issue de l'Affinity Benchmark [3] suggérée par F. Cazals : 1VFB_AB :C. Cette instance est composée de deux "patchs" correspondants aux sites de liaisons dont il faut mesurer la similarité. Le premier est composé de 220 atomes (19 résidus), le second de deux cent atomes (25 résidus). Nous avons obtenu un alignement de longueur 101 pour un RMSDc = 1.4579Å et un RMSDd = 1.20148. Ces résultats étaient concordants avec ceux de nos confrères. Par conséquent nous conservons ce modèle d'étude dans la liste des pistes d'exploitation.

2. INRIA Sophia Antipolis
3. `http://bmm.cancerresearchuk.org/~bmmadmin/Affinity/`

## 6.3 Discussion

La représentation de la comparaison de structures en graphe permet de modéliser simplement les différents appariements des structures. Nous avons créé quatre modèles opérationnels, du modèle résiduel standard au modèle atomique en passant par un modèle en FGS. La difficulté ici fut de rajouter des informations physico-chimiques. En effet, si des informations comme le critère d'appartenance à une structure secondaire est assez simple à choisir (les deux éléments appartenant ou non au même type de structure), cela est plus délicat pour d'autres critères comme l'hydrophobicité. Plus exactement c'est la pondération d'un critère par rapport à un autre qui est délicate. Nous avons essayé une méthode binaire où un seul manquement parmi la liste de critère invalidait l'appariement des éléments mais cela s'est avéré trop restrictif. De même nous avions étudié le problème en nous basant sur les matrices de BLOSUM 62 et 80, en choisissant une valeur qui validait ou non une paire mais lorsque nous avions confronté ce critère aux alignements de référence issus de SISY, nous avions observé que soit le seuil était très peu spécifique, soit pas assez et élaguait une partie des paires de référence. Ces résultats tendraient à dire qu'il vaut mieux se concentrer sur des critères de structures mais je les pondère en précisant que le nombre d'alignements de référence était faible (une trentaine) et que le seuil posé est fixe alors qu'il faudrait plutôt passer par un système de probabilités et de pondération au sein du graphe.

Les propriétés physico-chimiques et les fonctions de comparaison de base sont déjà implémentées, et ne nécessitent que des expériences pour déterminer la meilleure pondération possible. C'est pour cela que nous avons prévu des analyses à plus grande échelle, en nous basant sur plusieurs jeux de données comme MALIDUP, MALISAM, pour lesquels nous avons les alignements de référence, mais aussi en nous basant sur des alignements consensuels que nous créerons à partir des résultats d'un ensemble d'outils de comparaison dont entre autres TMalign, MICAN, Dalilite, et SANA.

L'un des moyens de contourner ce problème fut de passer au niveau des FGS, cela augmente la taille du graphe d'alignement mais enlève des contraintes d'appariements liées à la nature des résidus considérés.

L'autre grande force de Shinobi est la génération des arêtes en fonction de critères de distances, ces critères contraignent l'alignement final tant en distance absolue dans l'espace (ce qui permet de travailler sur de la similarité globale ou locale suivant les besoins) qu'au niveau de la déviation globale entre les éléments alignés (RMSDd).

## 6.4 Résumé du chapitre

Shinobi est un outil polyvalent de modélisation de comparaison de structures. Il permet de comparer des squelettes de protéines, en tenant compte ou non de la position des chaînes latérales mais aussi de comparer les protéines au niveau de leurs fonctions chimiques ou atomiques. De plus, la possibilité de choisir la dispersion autorisée des éléments, sur toute la protéine ou de manière plus locale, permet d'affiner la recherche. Ainsi cela peut être une comparaison globale des squelettes ou la recherche de sites de liaisons. L'outil, utilisé de manière conjointe avec Ninjas permet la recherche et la détection de sous-structures

communes renvoyées sous forme d'un ou plusieurs alignements. Ces alignements peuvent être composés de paires d'éléments uniques ou bien un élément peut être appareillé plusieurs fois. Ce dernier cas de figure a une application dans les perspectives de recherche de zones complémentaires entre surfaces protéiques.

# Chapitre 7

# Applications des outils dans la recherche de similarités au sein de structures : études de cas

## 7.1 Introduction

ShinobiNinjas (ShiNi) est un couple d'outils pensé pour rechercher des sous-structures similaires au sein de deux protéines. Pour le tester et vérifier sa qualité, nous avons commencé par l'utiliser sur une instance composée d'une protéine face à elle-même. Ce simple test a pour but de vérifier que l'outil détecte bien le meilleur alignement. Les résultats ont permis de vérifier que l'alignement optimal était bien trouvé et ont mené en plus à une étude sur les répétitions internes au sein des structures protéiques, décrite dans le chapitre 8. Nous avons ensuite testé notre outil sur des cas d'alignements de structures non linéaires : les cas de permutations circulaires et les cas de charnières. Les deux cas nécessitent un choix différent des paramètres d'entrée, car le premier cas sera directement inclus dans un alignement global, mais le second nécessite d'assembler des alignements puisque Ninjas n'est pas flexible. Enfin, nous avons observé la différence entre nos différents modèles ($C\alpha$, $C\alpha C\beta$et $FGS$) et nous nous sommes également intéressés aux résultats potentiels dans le cadre de la comparaison de surfaces.

## 7.2 Utilisation de ShinobiNinjas : modélisation d'une question biologique

A l'instar des autres outils, ShiNi s'exécute pour répondre à une question biologique spécifique. Notre outil n'est pas conçu pour estimer globalement la similarité de deux structures protéiques. Il permet de rechercher des sous-structures particulières communes à ces structures.

**Recherche de la plus longue sous-structure commune** Si le but est de trouver la plus grande sous-structure commune à deux structures, une analyse au niveau résiduel ($C\alpha$ ou $C\alpha C\beta$) est utilisée. Il est nécessaire de relâcher la contrainte de localité de Shinobi($\zeta = \infty$) pour permettre à l'outil de parcourir globalement la structure. Le choix de la flexibilité autorisée entre les paires de résidus alignés (utilisé pour les analyses standards effectuées dans ce chapitre) est établi à $\lambda = 3.0$Å après réalisation d'expériences. Il faut également choisir la robustesse des graines.

**Définition 7.1** *(Robustesse d'une graine) Une graine est dite robuste si les résidus sont en contacts ($\zeta_s$) et si la différence de distances entre paires de résidus est inférieure à $\lambda_s$.*

Les contacts selon la définition des études sur les cartes de contacts présentée dans la partie 2 correspondent à des distances entre éléments inférieures à $\zeta_s = 7.5$Å et la différence de distances entre paires de résidus à $\lambda_s = 2.0$Å. *Remarque* :

de récentes analyses préliminaires issues de la section d'identification de familles protéiques tendent à montrer qu'augmenter la distance de contact des graines à $10.0$Å pourrait accroître la précision des résultats.

La recherche étant ici géométrique, nous utilisons un filtre interdisant l'appariement de résidus issus d'hélices avec des résidus issus de feuillets.

Ce paramétrage permet, comme vont le montrer les résultats, de produire des alignements séquentiels ou non-séquentiels de deux structures en maintenant le RMSDc du dit alignement sous un seuil $\tau$ fixé par l'utilisateur.

**Alignement de structures flexibles** Lorsque la comparaison précédente de deux structures retourne un alignement partiel des deux structures, l'une des explications possibles est que l'une des structures possède une ou plusieurs charnières. Cette hypothèse se vérifie en réappliquant une comparaison en basculant en mode local (eg. en réduisant la valeur $\zeta$ à $15$Å - valeur utilisée par Probis pour créer l'environnement de recherche autour d'un $FGS$[64]) et en faisant retourner plusieurs résultats à ShinobiNinjas. Ces résultats correspondent à un ensemble de sous-structures locales similaires. Si l'outil retourne deux bons alignements non-chevauchants, alors une charnière a été détectée (de même pour trois alignements etc.)

**Alignement utilisant les groupes fonctionnels ($FGS$)** Les résultats montrent que pour les cas d'étude précédemment cités, une modélisation utilisant les groupes fonctionnels n'apporte rien directement et nécessite des ressources temps/machine supplémentaires pour un résultat équivalent. En revanche, cette modélisation trouve son intérêt dans la recherche très locale de sous-structures communes, ce qui a été montré par Konc et collègues dans [64]. Les structures ne sont plus observées dans leur globalité mais en surface et s'axent vers la détection de motifs très locaux comme les sites catalytiques ou les sites de liaison.

De même le modèle atomique se destine à la comparaison locale de surfaces protéiques.

## 7.3 Permutations circulaires

La principale difficulté de la détection de permutations circulaires au sein de structures protéiques réside dans l'indépendance de la séquence. Cela car cette contrainte usuellement utilisée permet d'éviter l'hyper-fragmentation des alignements. Un alignement fragmenté est un alignement contenant des sauts dans la séquence. Dans le cas d'un alignement non-séquentiel, l'outil va chercher une solution qui optimise sa fonction de score, fonction pénalisant plus ou moins les croisements. Cette section présente les résultats de ShiNi sur un jeu de données non-séquentiel et montre les forces et faiblesses de notre outil. Ces résultats remettent principalement en question le postulat de l'alignement optimal de deux structures, tel que l'a titré Adam Godzik : The structural alignment between two proteins : is there a unique answer ?[38].

Nous avons commencé par observer les résultats ( longueur et RMSDc des alignements) de ShinobiNinjas face à des résultats issus de la littérature ou calculés. Puis nous avons comparé les alignements issus de plusieurs outils plus en détails : TMalign, un outil séquentiel, MICAN, l'une des références des outils de détection des permutations circulaires, FlexSnap, un outil de chaînage d'alignements locaux qui permet la détection de charnières et est non-séquentiel donc devrait pouvoir détecter les permutations et ShinobiNinjas.

Les tests ont été effectués avec le mode $C\alpha$ (résiduel classique) pour rendre les méthodes comparables, nous avons néanmoins utilisé un filtre sur les structures secondaires et interdit l'appariement d'un résidu issu d'une hélice $\alpha$ avec un résidu issu d'un feuillet $\beta$.

### 7.3.1 Jeux de données : MALIDUP

Le jeu de données choisi (MALIDUP) [28] est une base de données composée de 241 alignement structuraux de domaines homologues. Ces alignements ont été effectués manuellement pour pallier les biais des outils d'alignements de séquences ou de structures. Minami et collègues [80] l'ont utilisé pour tester leur outil, MICAN, et ont artificiellement créé des alignements structuraux non séquentiels en permutant des portions de structures. Nous avons utilisé ce jeu de données non séquentiel (nommé MALIDUP-ns) car il présente plusieurs avantages dont le fait d'avoir des alignements de références manuels.

### 7.3.2 Résultats : MALIDUP-ns, détection de permutations circulaires

Nous avons comparé nos algorithmes avec ceux trouvés dans la littérature. Les résultats ci-dessous (tableau 7.1) sont une combinaison de résultats issus de [80] et de calculs effectués par nos soins.

Les résultats montrent logiquement que les deux méthodes séquentielles (DaliLite et TMalign) échouent en n'alignant qu'une partie des structures, ce car leur modèle, par construction, interdit les croisements au sein de l'alignement final. Cela s'observe au niveau des écarts entre les moyennes relatives aux longueurs des alignements Les résultats de FlexSnap sont surprenants car l'outil n'est pas contraint par l'ordre des résidus et peut donc théoriquement détecter les cas de permutations. De plus, l'outil a échoué dans 44 cas, ne produisant aucun résultat, et, même si ce mémoire n'est pas centré sur les performances

TABLE 7.1 – Qualité des outils d'alignements de structures sur le jeu de données MALIDUP-ns
Nali correspond au pourcentage moyen du nombre de résidus alignés divisé par la taille de la plus petite structure, RMSDc corespond au RMSDc moyen associé $N_{ali,outil-ref}$ mesure la différence entre les longueurs moyennes d'un outil et de la référence. De même, $RMSDc_{outil-ref}$ relate la différences entre le RMSDc moyen d'un outil et celui de la référence.

| Méthode | $N_{ali}$ % | RMSDc | $N_{ali,outil-ref}$ | $RMSDc_{outil-ref}$ |
|---|---|---|---|---|
| Référence[80] | 76.9 | 2.5 | - | - |
| DaliLite[48]* | 61.4 | 2.73 | -15.48 | 0.23 |
| TMalign[126]* | 60.5 | 2.94 | -16.38 | 0.44 |
| MICAN[80]* | 81.6 | 2.57 | 4.72 | 0.07 |
| DEDAL[33]* | 73.6 | 2.46 | -3.28 | -0.04 |
| SNAP[101]* | 72.9 | 2.01 | -3.98 | -0.49 |
| GANGSTA+[104]* | 73.3 | 2.47 | -3.58 | -0.03 |
| SCALI[122]* | 60.2 | 2.37 | -16.68 | -0.13 |
| MASS[34]* | 67.6 | 1.65 | -9.28 | -0.7 |
| SAMO[29]* | 77.5 | 2.84 | 0.62 | 0.34 |
| SANA[112] | 85.1 | 2.24 | 8.22 | -0.26 |
| CECP[17] | 70.2 | 3.57 | -6.68 | 1.07 |
| FlexSnap[102]** | 48.6 | 1.9 | -28.8 | -0.6 |
| ShiNi($C\alpha$) | 80.52 | 2.15 | 3.62 | -0.35 |

*données issues de [80]
** seuls 197 résultats sur 241 ont été obtenus avec FlexSnap, dans les autres cas, l'outil n'a pas été capable de trouver une solution.

des outils, il est à noter que FlexSnap est beaucoup plus lent que les autres heuristiques utilisées (instance comparée en une dizaine de minutes dans le pire cas observé contre moins d'une minute pour MICAN ou TMalign). MICAN et SANA ont produit des alignements en moyenne plus longs que les autres méthodes, avec même l'une des plus faibles moyenne de RMSDc pour SANA. Nous avons également calculé ces valeurs moyennes pour les alignements de référence et mesuré la différence entre les valeurs retournées par les outils et les références. ShiNi (avec le modèle basé sur les $C\alpha$) retourne des résultats proches de MICAN et des références, en effet notre outil réussit à détecter automatiquement les permutations. La proximité de nos résultats avec les références nous permet de conclure que ShiNi renvoie de bons alignements dans ce cas d'étude.

A l'aide de Samourai(présenté dans le chapitre 2), nous avons calculé d'autres scores correspondants aux alignements trouvés par quatre des outils cités : FlexSnap, MICAN, TMalign et ShinobiNinjas (figure 7.2) et nous avons également calculé ces scores en nous basant sur les alignements de références du jeu de données.

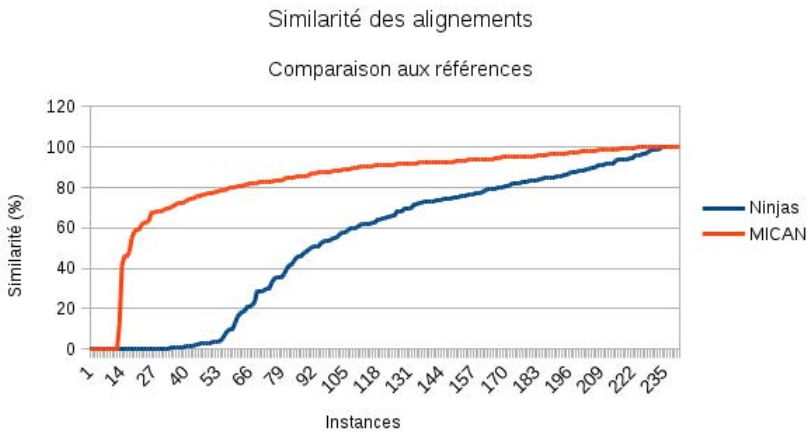



FIGURE 7.1 – Mesures de la similarité entre les alignements produits par MICAN et ShiNi avec les alignements de référence

Les sous-figures 7.2 (a) à (l) montrent la dispersion des scores des 241 comparaisons effectuées selon chaque outil ainsi que les références.

Globalement on distingue deux groupes : MICAN, ShiNi d'un côté, TMalign et FlexSnap de l'autre. Comme espéré ShiNi se comporte de manière assez similaire à MICAN, détectant correctement les permutations circulaires. TMalign renvoie aussi logiquement des résultats moins bons car il n'est pas adapté au jeu de données, de même ,comme dit plus haut, FlexSnap a retourné des résultats étonnamment mauvais. Si l'on s'intéresse de plus près aux résultats de ShiNi, on observe que, notamment pour la dispersion des valeurs, ses résultats sont très proches de ceux de la référence (voir en annexe pour les valeurs exactes). Les valeurs de RMSDc et RMSDd sont globalement concordantes entre MICAN, ShiNi et la référence même si l'on observe des valeurs maximales un peu plus élevées pour la référence. Dans la même lignée, les alignements des outils minimisent un peu plus le score SAS ce qui géométriquement est fort mais montre aussi que la réalité biologique diffère de l'alignement géométrique brut. L'observation des dispersions des identités de séquences (%) puis des alignements de ShiNi montrent que notre outil retourne des alignements pouvant être extrêmement fragmentés et par conséquent très peu similaires à l'alignement de référence. A l'inverse MICAN retourne des alignements très similaires à la référence (figure 7.1).

L'une des raison de cette hyperfragmentation réside dans le fait que ShiNi produit en premier lieu des alignements à appariements multiples et qu'ensuite, une implémentation de l'algorithme hongrois « choisit » l'un des ensembles uniques maximaux. Cet ensemble correspond à une solution optimale mais il est possible que d'autres solutions, moins fragmentées existent également.

### 7.3.3 Analyse de la fragmentation des alignements de ShiNi

Nous avons testé cette hypothèse en analysant les alignements à appariements multiples (noté **k-alignement**). La méthode est la suivante :

- recherche du plus long chemin du k-alignement à l'aide de l'algorithme hongrois (ce qui correspond au résultat initial de ShiNi).
- duplication du k-alignement (de taille k) en k alignements de longueur (k-1), une paire étant retranchée à l'alignement initial.
- pour chaque (k-1)-alignement, recherche du plus long chemin à l'aide de l'algorithme hongrois.
- si le chemin obtenu est de la même longueur que le résultat initial mais différent de ce dernier il est conservé.
- calcul des scores pour tous (k-1) alignements conservés.

Cette heuristique renvoie pour chaque (k-1) alignement une solution optimale qui n'est pas assurée d'être 1. la seule 2. la meilleure. Il faudrait une analyse exhaustive du k-alignement pour déterminer le meilleur alignement non redondant le moins fragmenté. L'objectif ici était de fournir une analyse préliminaire montrant qu'il existe plus d'une solution de longueur maximale et que le taux de fragmentation de l'alignement est variable. L'étude montre qu'il existe de nombreuses solutions de longueurs maximales pour chaque comparaison, toutes respectent les seuils donnés à ShiNi et sont donc toutes acceptables. En conclusion, une optimisation de ShiNi serait de modifier l'algorithme de recherche pour sélectionner l'alignement qui minimise la fragmentation (travaux en cours).

### 7.3.4 Observations sur les superpositions associées aux alignements

L'alignement est le résultat principal d'un outil de comparaison mais il est également possible de s'intéresser à un autre résultat qui est la superposition associée à l'alignement. L'hypothèse formulée ici est que deux outils retournant, de part leurs fonctions de scores différentes, des alignements différents, puissent retourner des superpositions relativement similaires. Pour tester cette hypothèse nous avons superposé les structures en nous basant sur les matrices de rotations associées puis, nous avons mesuré toutes les distances inter-résidus. La figure 7.3 représente toutes les distances inter-résidus triées après superposition optimale selon l'alignement de référence, MICAN, FlexSnap, TMalign et ShiNi. Ce graphique montre très nettement que les superpositions de la référence, MICAN et ShiNi retournent des distances pratiquement identiques. L'explication la plus probable est que les trois alignements correspondent à une superposition pratiquement identique. De même la table 7.2 résume les analyses partielles faites sur 665 comparaisons issues du jeu de données SKOLNICK, contenant 40 structures. On observe que les valeurs moyennes minimales, maximales et autres médianes sont pratiquement identiques, ShiNi ayant une valeur minimale moyenne plus faible que TMalign et MICAN. Ces résultats tendent à montrer que les outils convergent vers une superposition optimale alors que les scores associés à ces alignements divergent significativement (cf figure 7.4).

TABLE 7.2 – Moyennes des dispersions des distances issues des 665 instances de comparaison du jeu de données de Skolnick

Les distances inter-résidus après superposition selon MICAN, TMalign et ShiNi ont été mesurées puis leurs dispersions. Le tableau suivant est composé de la moyenne de chacune des catégories.

| Outil | MICAN | Tmalign | ShinobiNinjas |
|---|---|---|---|
| Minimum | 0.4563917383 | 0.4516774503 | 0.3184751429 |
| Quartile 1 | 14.2335469514 | 14.0341392571 | 13.7715693303 |
| Médiane | 19.6734633302 | 19.4169914549 | 19.0756706195 |
| Quartile 3 | 25.3754237536 | 25.0911075116 | 24.7013090395 |
| Maximum | 45.818067453 | 45.0061725316 | 45.4113358865 |

## 7.4 Détection de changements conformationnels, charnières Hinges

La flexibilité intrinsèque des protéines se traduit par des mouvements allant de la vibration d'un atome à de larges mouvements de chaîne. Cette dynamique des protéines joue un rôle important dans leur fonction. Des protéines homologues peuvent ainsi avoir des conformations différentes qu'il est important de pouvoir détecter. Les outils classiques comme TMalign ou MICAN sont basés sur une superposition rigide, séquentielle ou non, des structures et buttent ainsi sur cette flexibilité. Les outils dédiés comme Flexprot et FATCAT sont optimisés pour la recherche de flexibilités mais ont le défaut d'être séquentiels. FlexSnap est une solution à la fois flexible et non-séquentielle basée sur de l'assemblage de petits fragments bien alignés. Cette approche d'assemblage nous a donné l'idée de faire de même à partir des alignements obtenus avec ShiNi et les résultats se sont montrés prometteurs.

### 7.4.1 Méthodologie

La méthode débute par la création d'un graphe d'alignement basé sur les carbones centraux des acides aminés ($C\alpha$) dans lequel on interdit à un résidu appartenant à une hélice de se coupler avec un résidu issu d'un feuillet. Ce car les flexibilités recherchées ici sont de grands mouvements induits par des charnières (dans les boucles), les structures secondaires sont, elles, maintenues. Les différences de distances entre deux paires de $C\alpha$ sont tolérées à 4.0 Ået la recherche d'alignement peut s'effectuer sur l'intégralité des protéines ($\zeta = \infty$). En revanche les graines choisies sont des graines dites de contact (ne sont autorisées que des graines dont les $C\alpha$ sont éloignés au plus de $\zeta_s = 9.0$Å, avec une tolérance de $\lambda_s = 2.0$Å) La différence principale de cette méthode par rapport à d'autres (et c'est ici qu'intervient le rapprochement avec FlexSnap) est l'étude des n premiers résultats de Ninjas à partir du graphe créé. Ces résultats sont récupérés et assemblés si possible, la règle d'assemblage est un chevauchement maximal de 10%. Le plus long résultat est ensuite conservé et utilisé en comparaison.

### 7.4.2 Exemple : 1U42(A) versus 1U36(A)

Cet exemple, tiré de [102] illustre les différences entre alignements rigides (tmalign), hyperflexibles (apurva), flexibles (flexsnap) et mixtes (shinobi). La superposition rigide de **1U42** et **1U36** avec un outil rigide comme TMalign, MICAN ou encore DALI, retourne un alignement partiel, comme illustré par la figure 7.5, qui aligne la plus grande sous-structure commune. La charnière n'est pas prise en compte, par conséquent l'alignement est incomplet. Apurva, alignant les structures selon une somme de similarités locales, détecte un bon alignement global (couverture de la structure requête à 100 % et un score de l'outil égal à 0.79, pour une variation allant de 0 à 1). Tmalign, qui est un outil séquentiel rigide, retourne une comparaison qui recouvre la requête à moins de 60 %,pour un RMSDc égal à 2.0Ået un TMscore= 0.53. FlexSnap qui lui est un outil dédié à la recherche de flexibilités sous forme de charnières au sein des structures protéiques a aligné 100 résidus (soit la quasi totalité des structures) avec un RMSDc associé de 0.89Åen insérant une charnière. De même notre outil ShiNi a renvoyé deux alignements structuraux pertinents en premiers résultats : l'un comprenant 55 résidus pour un RMSDc de 0.55Ået l'autre de 45 résidus (RMSDc=0.81Å). L'analyse de ces deux alignements révèle qu'ils se chevauchent sur moins de 5% au niveau de la charnière. On en conclut que notre méthode permet de détecter les points de flexibilités en assemblant de bons alignements rigides locaux.

## 7.5 Discussion

Ce chapitre montre que ShiNi est un outil rivalisant, en termes de qualité, avec les autres outils disponibles. Il est capable de détecter des alignements non-séquentiels mais également flexibles. La contrainte de superposition ($\tau$) garantit la faible déviation globale après superposition et donc un bon alignement. ShiNi est un outil rigide qui permet de détecter des flexibilités en combinant les résultats d'une même instance.

Le nombre de résultats, d'alignements produits pour une même instance repose la question de l'alignement optimal. En effet, tous les alignements sortis sont géométriquement bons. Une amélioration, un tri dans ces alignements va être effectué pour minimiser le nombre de gaps dans l'alignement car ceux-ci sont considérés comme moins probables. Cela car un alignement hyper fragmenté sous entend une grosse recombinaison au sein des structures et à l'heure actuelle, aucune étude ne va dans ce sens. Par contre nous avons obtenu plusieurs alignements peu fragmentés pour une même comparaison et ainsi la question de l'alignement optimal ressurgit. Pour trancher entre ces alignements de ShiNi (entre eux et face à d'autres outils), nous avons implémenté Samourai, un outil qui, pour un alignement donné, calcule une dizaine de scores. Cela a montré que nos outils détectaient des alignements structuraux bons quel que soit le score. Comparer ces scores pour une même instance, sans que les alignements aient été optimisés pour ces scores, montre les limites des méthodes actuelles. Chacune est optimisée pour un score donné, ainsi une même comparaison peut donner des alignements, et donc des scores assez différents d'un outil à l'autre. La solution serait donc de choisir l'alignement qui est optimal pour un maximum de scores.

Mais, la comparaison des superpositions optimales associées à ces alignements montre

une perspective différente : certaines méthodes convergent vers la même superposition. Cela impliquerait que l'alignement n'est qu'une variable soumise à une fonction de score, les méthodes ont détecté la même similarité mais l'expriment différemment. Cette étude manque de tests statistiques à plus grande échelle pour réellement conclure sur ce sujet mais observer les superpositions des méthodes existantes pourrait permettre de catégoriser des méthodes différentes convergentes. Si cette hypothèse est avérée, il va être nécessaire de déterminer un nouvel angle d'approche pour créer l'alignement optimal à partir de la superposition. Une possibilité est de maintenir l'effet de seuil de déviation en sélectionnant toutes les paires de résidus à moins de $\tau$Å et en cherchant au sein des ces paires l'alignement minimisant les sauts.

## 7.6 Résumé du chapitre

La compréhension des relations structurelles (au sens similarités) de deux protéines ne peut se limiter à une simple recherche de la sous-structure commune la plus grande. Une étude séquentielle peut bloquer sur les cas de permutations circulaires et une superposition rigide ignorera les flexibilités. Or ces différences que l'on peut considérer comme mineures (deux structures pouvant être identiques à une charnière près) sont essentielles et doivent être prises en compte. Nos méthodes basées sur la recherche d'alignements structuraux locaux rigides (séquentiels ou non) permettent de détecter les différents cas de figure et ainsi de couvrir l'ensemble des possibilités connues actuellement. Nous pouvons détecter les sous-structures communes "classiques" (séquentielles rigides), les cas de permutations circulaires, ainsi que les flexibilités liées aux charnières. Ces détections ne dépendent que du paramétrage effectué en amont de l'analyse et nous avons présenté ici des exemples permettant d'illustrer nos méthodes de recherches des différents cas ainsi que l'explication des résultats renvoyés par nos outils. ShinobiNinjas (ShiNi) est donc un outil polyvalent permettant de comparer deux structures protéiques en définissant une question biologique au préalable.

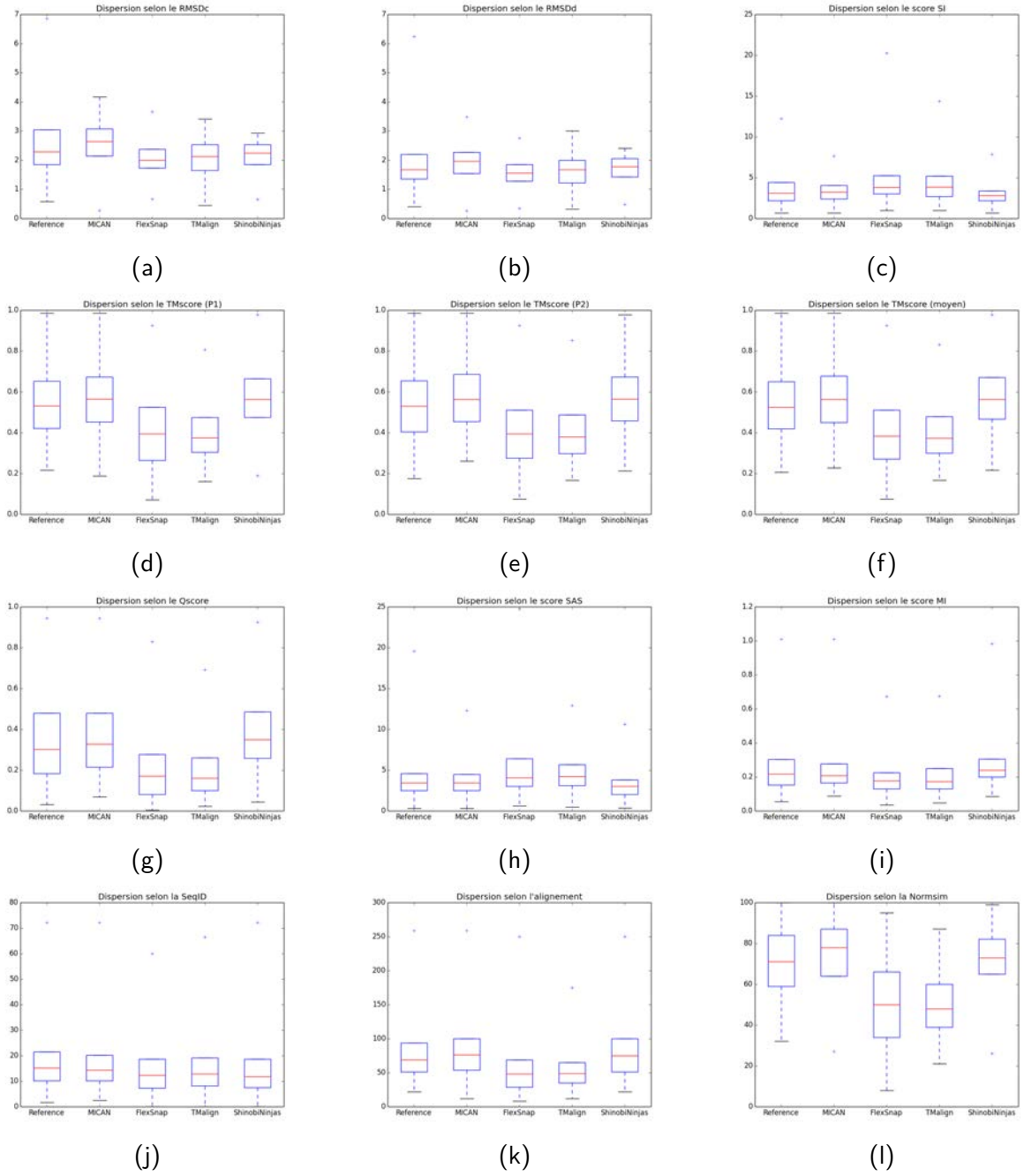



FIGURE 7.2 – Dispersion des scores des alignements issus de FlexSnap, MICAN, TMalign et ShinobiNinjasainsi que les scores basés sur les alignements de référence.
Les scores et valeurs suivantes sont à maximiser : ALI : longueur de l'alignement, MI : Score MI, Qscore, TMscore (TMP1/TMP2 : TMscore relatif à la longueur de la protéine P1 (resp. P2), TMmoy : TMscore relatif à la longueur moyenne des protéines), Normsim : Valeur de similarité normalisée, Seqid : pourcentage d'identité de séquence.
Les valeurs de RMSDc, RMSDd, SI, SAS sont considérées meilleures quand minimisées

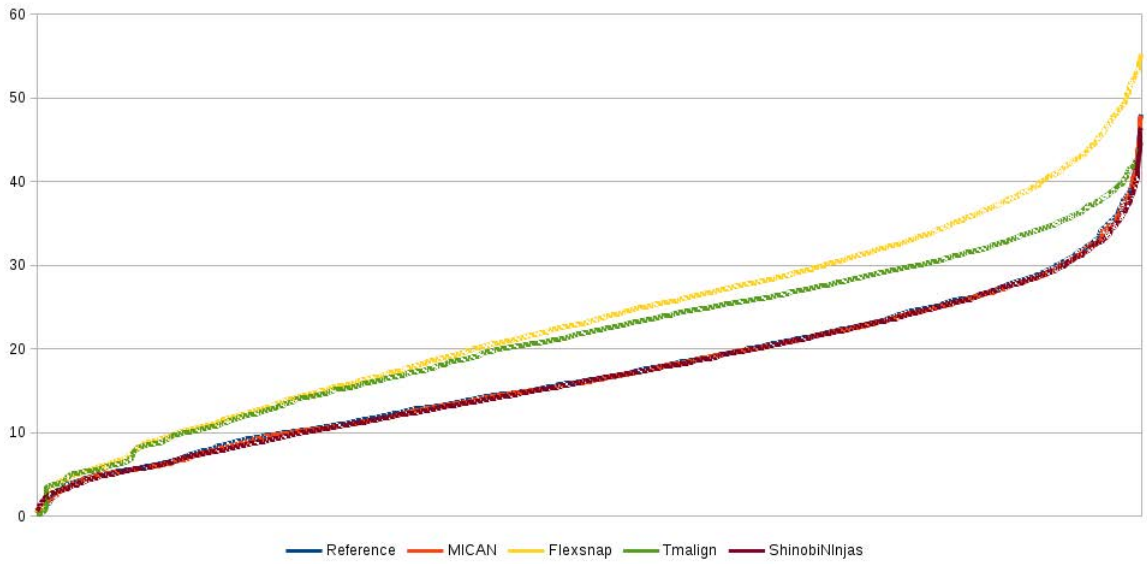



FIGURE 7.3 – Distances inter-résidus après superposition du domaine **d1a4pa_**

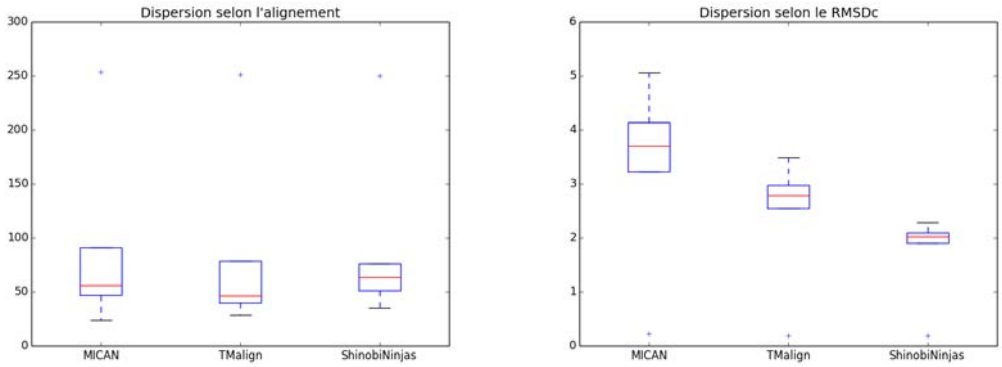



FIGURE 7.4 – Dispersion des alignements et des RMSDc associés en fonction de l'outil de comparaison.

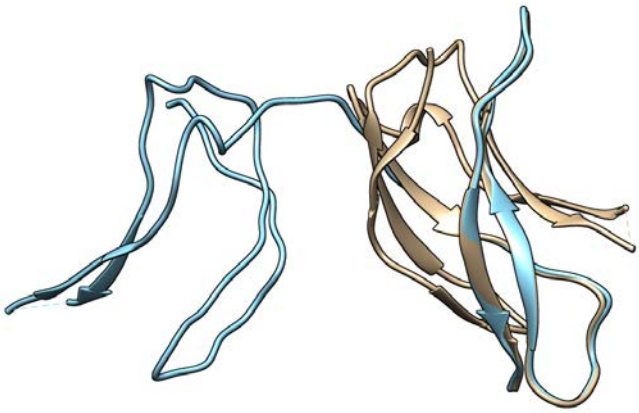


FIGURE 7.5 – Superposition rigide de **1U42** et **1U36** via Chimera, la charnière est ignorée et l'alignement correspondant est donc partiel

# Chapitre 8

# Détection automatique de répétitions structurales au sein des protéines

## 8.1 Introduction

Un des premiers résultats de ShinobiNinjasfut un test de comparaison d'une protéine sur elle-même dans le but de vérifier que l'outil détectait correctement l'alignement optimal d'une structure avec elle-même. Nous avons donc utilisé **2BNH,A**, une protéine de 456 résidus et l'avons comparée avec elle-même en nous basant sur ses $C\alpha$et récupéré les 40 meilleurs alignements. 17 furent supprimés car présentaient un pourcentage de gap supérieur à 50 %. Les autres furent intégrés à la matrice de la figure 8.1. On observe que, à 4.0Å, de nombreux résidus s'alignent avec une quinzaine d'autres résidus en plus d'eux-mêmes. La figure présente tout d'abord la diagonale qui correspond à l'alignement des résidus avec eux-mêmes. Cela montre que ShiNi détecte bien le meilleur alignement d'une protéine avec elle même. Puis on remarque de nombreux alignements en parallèle. Ces alignements correspondent au motif répété qui compose la protéine solénoïde. A partir de ce constat, nous avons cherché à savoir si ShinobiNinjaspouvait être utilisé dans le cadre d'un protocole de détection automatique de ces répétitions internes.

La comparaison d'une structure avec elle-même pour notre outil est une tâche longue. De plusieurs heures pour la version initiale de Ninjas(accélérée depuis), de plus s'est posé le problème de la découpe des alignements. En effet, un alignement ne correspond pas à un motif structural répété mais à une sous-structure présente deux fois. Cette sous-structure peut être composée du motif seul mais les longs alignements sont composés, notamment dans le cas de l'exemple, par plusieurs motifs successifs. Le problème de la découpe des alignements étant encore un sujet ouvert. Nous avons testé une version plus simple de la détection de structures répétées avec une méthode naïve qui consiste à créer des motifs structuraux continus dans la séquence. Ces motifs servent de dalles et sont ensuite chaînés pour tenter de reconstruire la protéine selon une couverture donnée.

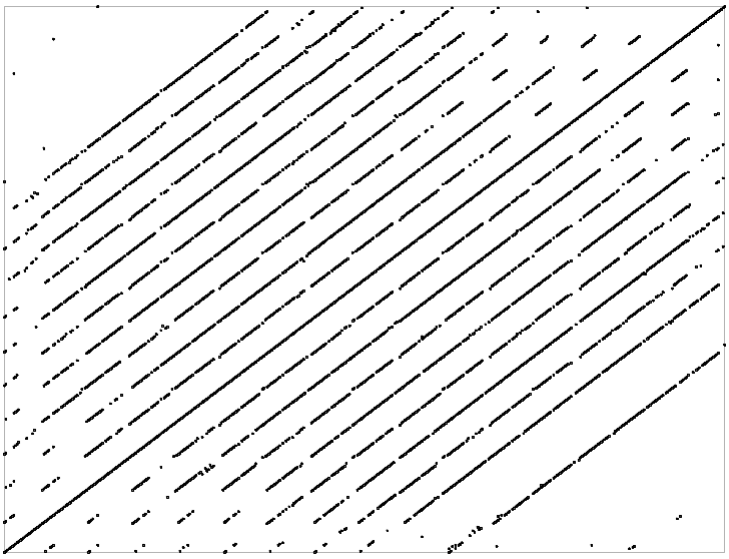


FIGURE 8.1 – Matrice modélisant l'apparition de paires de résidus alignés dans les alignements de **2BNH,A**. Un point en position $[i, j]$ de la matrice correspond à l'appariement des $i^e et j^e$ résidus de la protéine.

Dans ce chapitre nous présentons une version simplifiée du graphe d'alignement, nommé *graphe de dalles* qui découpe une protéine en fragments continus et relie les fragments indépendants. Ce graphe est ensuite parcouru pour y détecter tous les ensembles de dalles similaires correspondant à des répétitions internes.

Notre exemple suivi **2BNH,A** est une protéine inhibitrice de la ribonucléase chez le porc constituée de 456 résidus. C'est un solénoïde, plus précisément une protéine à répétitions riches en leucine, LRR (leucine-rich repeats). Sa forme en fer à cheval (figure 8.4) se décompose en agencements d'hélices $\alpha$ et de feuillets $\beta$ répétés. Nous avons sélectionné cette protéine pour illustrer ce chapitre car plusieurs résultats sont disponibles dans la littérature : Parra *et al.* [95] ont dénombré 8 répétitions d'un élément contenant 57 résidus tandis que Murray *et al.* en ont trouvé entre 15 et 16 avec leur outil DAVROS [81] et à l'aide de l'outil ConSole de Hrabe et Godzik [53] nous avons détecté 15 répétitions de longueur 28 au sein de cette protéine.

Ces travaux ont donné lieu à une publication dans la conférence JOBIM 2015 [70] (publication numéro 2.).

## 8.2 Modélisation simplifiée d'un protéine : Graphe de dalles

Notre méthodologie repose principalement sur la modélisation d'une protéine $(P)$ sous forme d'un *graphe de dalles*, initialement nommé graphe de fragments dans l'article [70].

**Définition 8.1 (Dalle de taille $k$ d'une protéine)** *Motif structural correspondant à un ensemble de $k$ résidus continus dans la séquence d'une protéine.*

**Définition 8.2 (Graphe de dalles de taille $k$)** *Un graphe de dalles est un graphe G=(V,E) avec $V_G$ l'ensemble des dalles de longueur $k$ et $E_G$ l'ensemble des arêtes liant les dalles compatibles. Deux dalles sont compatibles si elles sont de même longueur et se superposent avec une valeur de déviation faible.*

Soit une protéine $P$ de longueur $N$ et $k$ une taille de dalle donnée, le graphe de dalles G=(V,E) associé contient $N - k$ dalles chevauchantes (cf figure 8.3). La constitution des dalles s'effectue par la découpe de la protéine en fragments continus de longueur $k$, chaque fragment étant ce que l'on nomme ici une dalle (figure 8.2).

FIGURE 8.2 – Modélisation d'une protéine en dalles et graphe de dalles associé
Découpage d'une protéine en fragments continus de taille $k$ (dalles)

FIGURE 8.3 – Graphe de dalles
Une arête relie deux sommets (représentés par les rectangles bleus) si les dalles correspondantes sont compatibles.

Chacune de ces dalles correspond à un sommet $v$ du graphe G=(V,E). Il existe une arête $e_{v,w} \in E_G$ entre deux sommets $v, w \in V_G$ si et seulement si les dalles associées sont *compatibles*.

**Définition 8.3 (Compatibilité de dalles)** *Deux dalles sont compatibles si et seulement si :*
- *Elles sont non-chevauchantes.*
- *Le RMSDc associé à la superposition optimale des dalles est inférieur à un seuil $\tau$*

La valeur de $\tau$ est un paramètre utilisateur (par défaut, $\tau = 3.0$Å).

L'algorithme suivant (5) résume le procédé de création du graphe de fragments par Kunoichi.

**Algorithme 5** Création du graphe de fragments par Kunoichi

**Require:** $P(N)$ ▷ Une protéine de taille $(N)$
**Require:** $\tau$ ▷ valeur limite de RMSDc entre fragments autorisée
**Require:** $f_{size}$ ▷ Longueur des fragments créés
$Residues = \{Residue_1, \ldots, Residue_N\}$ ▷ Ensemble des résidus de la protéine
$Fragments = \{\}$ ▷ Ensemble des fragments continus
**for** $i = 0; i < |Residues|; i++$ **do**
$CurrentFragment = Residues[i : i + f_{size}]$ ▷ Création du fragment commençant au $i^{\text{ème}}$ résidu et de longueur $f_{size}$
$Fragments = Fragments \cup \{CurrentFragment\}$ ▷ Ajout du fragment courant
**end for**
$Edges = \{\}$ ▷ Ensemble des arcs
**for** $f = 0; f < |Fragments|; f++$ **do**
**for** $g = f + 1; g < |Fragments|; g++$ **do**
**if** compatibility($Fragment, AnotherFragment$) $==$ true **then**
$edge = (Fragment, AnotherFragment)$ ▷ Création d'un arc entre les deux fragments
$Edges = Edges \cup \{edge\}$
**end if**
**end for**
**end for**
**return** Graphe $(Fragments, Edges)$

## 8.3 Parcours du graphe de dalles

Le parcours du graphe s'effectue avec un solveur de cliques. La nécessité ici est de trouver l'ensemble des cliques de taille $R$. Une clique de taille $R$ correspond à un ensemble de $R$ dalles compatibles. Soit, selon nos critères, $R$ dalles structurellement similaires et non-chevauchantes. Cela correspond à une dalle, un motif structural, répété $R$ fois. L'ensemble des cliques de taille $R$ correspond donc à l'ensemble des motifs structuraux répétés $R$ fois au sein de la protéine. Par construction, notre méthode garantit que toutes les dalles sont structurellement similaires à $\tau$Å près. Pour nos études, nous avons utilisé cliquer d'Ostergaard qui retourne les cliques souhaitées.

## 8.4 Analyse des ensembles obtenus

Lorsque le solveur a parcouru l'intégralité du graphe, il renvoie, ou non, une ou plusieurs cliques de taille $R$.

Soit G=(V,E) un graphe de dalles de longueur $k$ résolu avec un solveur pour un nombre de répétitions $R$ donné. Les trois résultats possibles sont :

1. Aucune clique. Cela signifie que dans $g$, il n'existe aucun motif structural (dalle) de longueur $k$ répété $R$ fois.
2. Une seule clique. Cela correspond à un motif structural de longueur $k$ répété $R$ fois.
3. Plusieurs cliques, soit plusieurs motifs structuraux répétés.

Le dernier cas est lui-même composé de deux types de cas. Il peut s'agir du même motif avec un décalage au niveau des extrémités, une légère variabilité faisant que l'algorithme trouve le même motif et renvoie ainsi les dalles alternatives. Se pose alors la question du choix du meilleur ensemble de dalles. Tous répondent aux critères exigés de part la construction du graphe, par conséquent nous avons cherché d'autres critères d'évaluation. Nous avons sélectionné une mesure, un score de similarité des structures secondaires moyen. Le pourcentage d'identité de structure secondaire de chaque paire de dalles d'un même ensemble est calculé et le pourcentage moyen déduit ($\overline{SSE}(f, f')$).

Le pourcentage d'identité de structure secondaire (SSE) pour deux fragments $f, f' \in F$ est défini par l'équation suivante :

$$SSE(f, f') = \frac{\sum_{i=1}^{n}(SSE(f_i, f'_i))}{N} * 100 \tag{8.1}$$

avec $SSE(f_i, f'_i) = 1$ si les résidus $f_i$ et $f'_i$ appartiennent à la même structure secondaire, $0$ sinon.

Ensuite, nous calculons dans un second temps le pourcentage d'identité de séquence moyen.

**Définition 8.4 (Meilleur ensemble de dalles)** *Soit $D = d_1, d_2, \ldots, d_n$ un ensemble d'ensembles de dalles. Le meilleur ensemble de dalles est l'ensemble ayant le pourcentage d'identité de SSE moyen le plus élevé et, structure secondaire moyen et, en cas d'égalité, le meilleur pourcentage d'identité de séquence moyen.*

L'autre possibilité en cas de multiples cliques est la découverte de deux motifs distincts. Par exemple, **2BNH,A** est composée d'une alternance de feuillets et d'hélices, si l'on sélectionne une taille de dalle assez faible, l'algorithme retourne dans ses résultats l'ensemble des feuillets dans une clique et l'ensemble des hélices dans l'autre.

## 8.5 Recherche orientée de répétitions structurales au sein des protéines

La recherche de répétitions est ici restreinte à la recherche d'une dalle, qui, répétée x fois, va constituer la protéine. Par conséquent, à la taille des dalles et au nombre de répétitions s'ajoute un troisième paramètre : la *couverture* $C$.

**Définition 8.5 (Couverture (des dalles))** *La couverture est la proportion de la protéine que l'ensemble de dalles similaires doit constituer.*

La couverture sert à donner un certain de tolérance à l'algorithme. Cela laisse la possibilité qu'une répétition ait été dégradée, qu'il y ait une petite flexibilité dans les motifs ou tout simplement que la protéine ne soit pas entièrement constituée d'un unique motif répété.

Le protocole de recherche orientée s'applique dans un contexte où l'on connait par avance le nombre de répétitions ($R$) recherchées( via la littérature ou encore un outil), il est également nécessaire de fournir le pourcentage minimal de recouvrement ($C$) de la protéine attendu.

### 8.5.1 Méthode

Le protocole utilise en premier lieu la longueur de la protéine, le nombre de répétitions recherchées et le pourcentage minimal de recouvrement pour calculer un intervalle de longueurs de dalles candidates. La longueur des dalles $k$ est contrainte par l'inégalité suivante :

$$\lceil \frac{C * N}{R} \rceil \leq k \leq \lfloor \frac{N}{R} \rfloor \tag{8.2}$$

avec $k = \lceil N/R \rceil$ la longueur maximale que la dalle correspondant à l'unité répétée peut atteindre et $k = \frac{C*N}{R}$ la longueur minimale des dalles pour couvrir $C\%$ de la protéine une fois assemblés. Le protocole lance Kunoichisuivi du solveur et du module d'analyses pour la taille de dalle la plus élevée et en cas d'absence de solution recommence en diminuant la longueur du dalle jusqu'à obtention d'une solution ou avoir parcouru l'ensemble des longueurs de dalles possibles comme le montre l'agorithme 6.

### 8.5.2 Résultats de la recherche orientée

Nous nous sommes basé sur les résultats préliminaires de ShiNi sur **2BNH,A** qui montraient 15/16 répétitions et la littérature qui en dénombrait huit ([95]) pour définir notre paramètre $R$. De même nous avons défini la couverture minimale )à 80% et une valeur de RMSDc $\tau$ variant entre 3.0 et 5.0Å.

**Algorithme 6** Recherche simple de répétitions avec Kunoichiet un solveur

**Require:** $R$ ▷ Nombre de répétitions recherchées
**Require:** $P(N)$ ▷ Une protéine de taille $(N)$
**Require:** $\tau$ ▷ valeur limite de RMSDc entre dalles autorisée
  $k_{max} = \frac{N}{R}$ ▷ Taille maximale des dalles autorisés
  $k_{min} = \frac{C*N}{R}$ ▷ Taille minimale des dalles autorisés
  **for** $k = k_{max}; k \geq k_{min}; k--$ **do**
    graphe_courant=Kunoichi$(P, R, \tau, k)$ ; ▷ Création du graphe de dalles avec Kunoichi
    ▷ Recherche de toutes les cliques $(all)$ maximales $(max)$ de taille au moins égale à $R$ $(R)$ dans le graphe courant
    clique_set=(solve($all, max, R,$ graphe_courant) ▷ stockage des cliques trouvées
    **if** is_empty(clique_set) $==$ false **then**
      **return** clique_set and $k$ ; ▷ retourne les cliques trouvées pour une taille de dalle $k$
    **else**
      continue ;
    **end if**
  **end for**

TABLE 8.1 – Recherche orientée de répétitions chez **2BNH,A**

| Nb répétitions | $\tau$(Å) | Couverture (%) | Nb résultats trouvés | Taille de dalles |
|---|---|---|---|---|
| 8 | 3.0 | 80 | 0 | - |
| 8 | 4.0 | 80 | 1 | 48 |
| 8 | 5.0 | 80 | 2 | 55 |
| 15 | 3.0 | 80 | 9 | 27 |
| 16 | 3.0 | 80 | 2 | 24 |

La recherche de 8, 15 et 16 répétitions avec Kunoichi(via l'algorithme simple 6) a retourné les résultats décrits dans le tableau 8.1. Les résultats montrent qu'avec un seuil $\tau = 3.0$Å, Kunoichine trouve aucune clique, aucun ensemble de dalles recouvrant au moins 80% de la structure. En revanche, si on augmente ce seuil à 4.0Å, Kunoichitrouve une dalle, de longueur 48 (soit composée de 48 résidus successifs), correspondant à ces critères.

Le nombre de répétitions et la couverture définissent un intervalle discret de taille de dalles. Pour 15 répétitions, cet intervalle varie entre 30 et 25. Les premières recherches à 3.0Å $(pour k = 30, 29, 28)$ furent infructueuses mais pour $k = 27$ le protocole a bien trouvé 15 dalles superposables à $3.0$Å visualisées sur la figure 8.4.

On remarque qu'avec un seuil de RMSDc de superposition de dalles assez faible (3.0 Å), Kunoichine retrouve pas les résultats de Parra *et al.* (8 répétitions), cependant, en augmentant ce seuil d'un point les résultats concordent mieux même si la longueur de dalles continue à différer (57 pour [95] contre 48 pour nous). Cela s'explique par une flexibilité plus importante de leur outil, flexibilité que Kunoichipeut exprimer si l'on augmente son seuil de

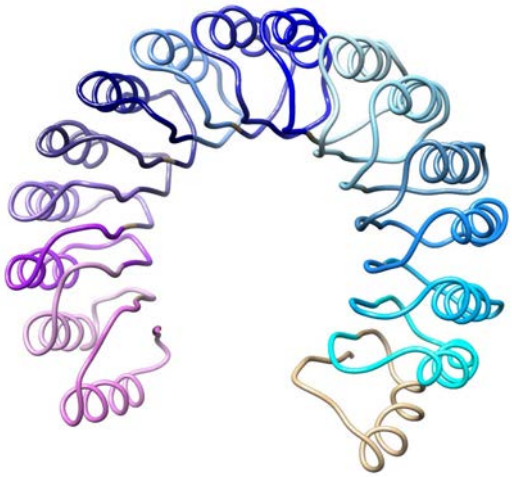


FIGURE 8.4 – Représentation des 15 dalles au sein de **2BNH**.
En dégradé du bleu au rose sont représentés les dalles (un par couleur), en beige on retrouve les résidus qui n'appartiennent à aucune répétition.

superposition.

Et en effet si le seuil est placé à 5.0 Å la longueur de dalles obtenue est de 55 résidus. De même lors de la recherche de 15 et 16 répétitions au sein de la structure.

On notera que dans plusieurs cas, Kunoichiretrouve plusieurs résultats, la figure 8.5 montre les différents pourcentages moyens d'identité de séquence et de structure secondaire pour chaque résultat. Au niveau des structures secondaires, le pourcentage moyen est entre 75 et 78% tandis que les pourcentages moyens d'identité de séquences sont faibles (environ 30%). Les résultats sont très similaires, au niveau des valeurs, le numéro 7 a un $\overline{SSE_{id}} = 74.62\%$ (contre 74.52% pour les deux résultats suivants) et un $\overline{Seq_{id}} = 29.45\%$ ce qui en fait le meilleur selon notre protocole et ce bien que son pourcentage moyen d'identité de séquence ne soit pas le plus grand.

### 8.5.3 Discussion

La principale faiblesse de cet algorithme (discutée plus longuement dans la section 8.6.3) est la nécessité de connaître à l'avance le nombre de répétitions recherchées. C'est un gros point noir car dans une utilisation concrète il est impossible de connaître à l'avance le nombre de répétitions recherchées. Mais ce protocole avait pour but de tester les performances, en terme de qualité de résultats, de la construction et de la résolution du graphe. Et dans ce cadre, les résultats sont encourageants, la recherche de cliques dans un graphe de dalles permet de retrouver le motif structural répété composant la protéine.

En conséquence, nous avons conservé la base de cet algorithme en relâchant la contrainte causée par le nombre de répétitions via un algorithme de détection *de novo* de répétitions.

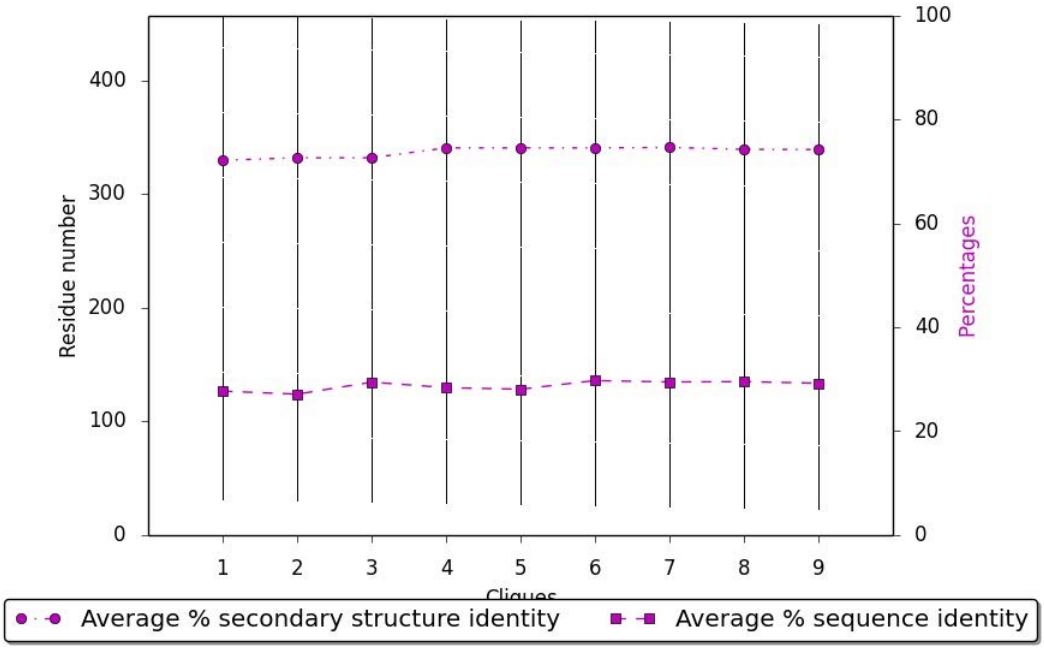



FIGURE 8.5 – Analyse des neuf cliques trouvées chez **2BNH,A** Avec $tau = 3.0$Å, une couverture = 80% et un nombre de répétitions égal à 15. Les pointillés noirs correspondent aux résidus appartenant à chaque clique.

## 8.6 Détection *de novo* de répétitions dans les structures protéiques

Le relâchement de la contrainte sur le nombre de répétitions passe par une recherche exhaustive incrémentale. Le protocole précédent est lancé plusieurs fois en modifiant ce paramètre $R$. Cette approche incrémentale se rapproche de la méthode de Parra *et al.* dans le sens où nous allons retomber sur des « tuiles » répétées recouvrant la protéine. La différence majeure est que nos dalles sont non-chevauchantes, contrairement à leurs tuiles (d'où la différence de dénomination). Nous allons donc tester ce protocole sans à priori sur le nombre de répétitions recherché.

### 8.6.1 Méthode

Le nombre de répétitions recherchées est initialisé à 2 et augmenter d'un cran jusqu'à ce que la taille minimale imposée sur les dalles soit atteinte. En effet, nous imposons qu'une dalle soit au minimum de taille 8 (paramétrable) pour être acceptée. Comme à chaque début d'analyse, l'outil défini les tailles de dalles possibles en fonction du nombre de répétitions souhaité et de la couverture demandée, l'algorithme s'arrête lorsqu'il est impossible de trouver x répétitions de dalles de taille supérieure à 8 suivant une couverture minimale $C$. Pour chaque

nombre de répétitions testé, nous allons créer les dalles de taille correspondante et vérifier leur similarité. Ce protocole est présenté par l'algorithme 7.

**Algorithme 7** Détection automatique de répétitions avec Kunoichiet un solveur

**Require:** $P(N)$ ▷ Une protéine de taille $(N)$
**Require:** $\tau(défaut : 3.0Å)$ ▷ valeur limite de RMSDc entre dalles autorisée
**Require:** $C(défaut : 80.0\%)$ ▷ Taux de recouvrement de la protéine par les répétitions
**Require:** $k_{size}(défaut : 8)$ ▷ Taille minimale absolue des répétitions
  $r_{max} = \frac{N}{f_{size}}$ ▷ Calcul du nombre maximal de répétitions recherchées
  **for** $r = 2; r \leq r_{max}; r++$ **do** ▷ Recherche de cliques de cardinal deux à $r_{max}$
    $k_{min} = \frac{N*C}{r}$ ; ▷ Calcul de la taille minimale des dalles pour $r$ répétitions
    **if** $k_{min} < k_{size}$ **then**
      $k_{min} = k_{size}$ ; ▷ Restriction de la taille limite au paramètre utilisateur
    **end if**
    $k_{max} = \frac{N}{r}$ ▷ Longueur maximale des dalles pour $r$ répétitions
    **for** $k = k_{max}; k \geq k_{min}; k--$ **do**
      graphe_courant=Kunoichi$(P, r, \tau, k)$ ;
      clique_set=(solve$(all, max, r,$ graphe_courant)
      **if** is_empty(clique_set) $==$ false **then**
        save(clique_set, $k$,$r$) ; ▷ Stockage des cliques trouvées
        break ;
      **else**
        continue ;
      **end if**
    **end for**
  **end for**

Ce nouveau protocole retourne l'ensemble des répétitions acceptables (c-à-d répondant aux critères donnés) d'une protéine.

### 8.6.2 Résultats

Nous avons testé le protocole décrit en 7 avec un recouvrement à 80.0 % et un RMSDc de superposition de dalles à 3.0Å sur **2BNH,A**. La recherche de répétitions est ici automatique, par conséquent nous allons effectuer une décomposition de la protéine, tous les niveaux de répétitions vont ressortir. Le tableau 8.2 résume les résultats obtenus, Kunoichidétecte plusieurs niveaux de répétitions, la protéine est presque parfaitement recouverte par deux dalles identiques avec une déviation à 3.0Ået peut se découper en plus petits dalles tout en gardant une bonne couverture. Au maximum 16 répétitions sont possibles avec nos critères, après soit la couverture n'est plus assez grande, soit la longueur des dalles est trop petite. D'autres répétitions peuvent exister mais à un niveau plus local, ici le protocole ne permet de découvrir que les dalles qui recréent la protéine lorsque mis bout à bout.

TABLE 8.2 – Détection de répétitions à 3.0 Å chez **2BNH,A**

| Taille des dalles | Nombre de répétitions | Pourcentage de recouvrement |
|---|---|---|
| 225 | 2 | 98.6842105263 |
| 141 | 3 | 92.7631578947 |
| 102 | 4 | 89.4736842105 |
| 84 | 5 | 92.1052631579 |
| 56 | 7 | 85.9649122807 |
| 27 | 14 | 82.8947368421 |
| 27 | 15 | 88.8157894737 |
| 24 | 16 | 84.2105263158 |

### 8.6.3 Discussion

Nous avons donc créé une méthode entièrement automatique de détection de répétitions structurelles au sein des protéines. La décomposition de la protéine en dalles permet d'identifier tous les motifs structuraux qui, répétés, recouvrent la dite protéine. Néanmoins, cette méthode a quelques lacunes clairement identifiées :

- Kunoichitravaille entièrement avec des dalles continus ce qui ignore les cas de répétitions dans lesquelles se trouvent des insertions/délétions de résidus. Les cas de permutations circulaires sont également ignorés.
- Lorsque cliquer renvoie une réponse positive, il se peut qu'il trouve plusieurs cliques et donc potentiellement différents ensembles de répétitions. Il est nécessaire de comparer ces différentes ensembles, par l'intermédiaire de scores (via Samouraipar exemple) mais également au sein de chaque ensemble. Un alignement multiple ( de séquence comme de structure ) est envisagé pour mieux étudier ces répétitions.

## 8.7 Discussion, perspectives

Comme tous les autres outils que nous avons développés, Kunoichirepose sur la complémentarité d'un graphe (ici un **graphe de dalles** ) modélisant une problématique biologique et un solveur de graphe. Ici le solveur est cliquer d'Ostergaard qui recherche des cliques au sein de graphes non-orientés. Les protocoles développés montrent que la recherche d'ensembles de dalles non chevauchants de protéines, sous la forme de recherche de cliques dans un graphe, permet de retrouver le motif de structure composant la protéine. La limite principale de ce modèle est bien entendu le fait que les dalles créées soient continues. Cela impose de fortes contraintes sur le graphe car impose une séquentialité au modèle, or, suite à des évènements évolutifs, des portions de la protéine peuvent migrer, être supprimées (délétions) ou de nouveaux résidus peuvent s'insérer. C'est typiquement le cas de l'anhydrase gamma carbonique **1QRL** qui possède une longue boucle en plein milieu des $\beta$-solénoïdes comme le montre la figure 8.6.

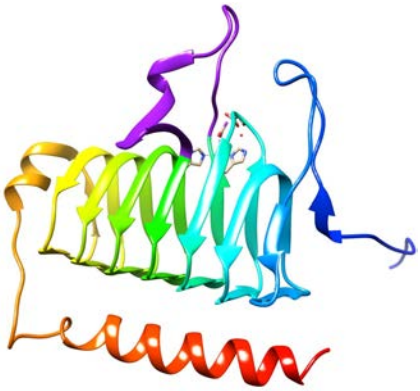


FIGURE 8.6 – Représentation 3D de **1QRL**, en violet la boucle insérée au sein de la structure répétée

Les dalles étant créés selon une fenêtre coulissante, la structure répétée est cassée et ne se retrouvera donc pas dans le graphe. Par conséquent, cette sous-structure ne sera pas ensuite détectée comme répétition. Plusieurs pistes sont envisagées afin de pallier cette lacune, l'une d'elle étant de combiner Kunoichià Shinobiet Ninjaslors de la création de dalles. Les dalles continus seraient alignés avec ninjas puis transformés en dalles discontinus (déjà liés les uns autres). Et c'est dalles discontinus reliés seront ensuite analysés par cliquer pour trouver le plus grand ensemble compatible comme dans la version classique de Kunoichi.

## 8.8 Conclusion, résumé du chapitre

La recherche de dalles continus, de dalles structurales, pour identifier des protéines solénoïdes s'est avérée efficace. Néanmoins la contrainte de continuité empêche la détection de répétitions internes dans les cas où un événement évolutif aurait induit l'insertion de résidus au sein d'une dalle. Au vu des résultats préliminaires de la comparaison d'une protéine avec elle-même, nous envisageons donc de remplacer le module de création de dalles par les résultats de ShinobiNinjassur cette instance. C'est-à-dire utiliser les alignements obtenus lors de la comparaison de la protéine avec-elle même comme dalles de base et chercher ensuite à les chaîner. Cela équivaut à certaines méthodes d'alignements de structures qui cherchent de petits alignements locaux similaires puis les chaînent pour former l'alignement final mais appliqué à la reconstruction de structures répétées. La détection de structures répétées au sein d'une protéine non solénoïde est différente car il n'est plus question de reconstruire la protéine mais de trouver quelques sous-structures similaires. Pour cela nous allons appliquer un protocole consistant à récupérer plusieurs alignements locaux de la protéine sur elle-même. Ces alignements correspondront à des motifs structuraux présents plusieurs fois dans la structure donc des motifs répétés.

# Chapitre 9

# Recherche de divergences entre structures fortement similaires

## 9.1 Introduction

Les protéines, particulièrement les enzymes, portent une ou plusieurs fonctions, opérationnelles ou non. Certaines enzymes, familles d'enzymes, ont des structures très similaires, des fonctions similaires, mais pas identiques. L'un des but de la biologie structurale est de comprendre le fonctionnement d'une enzyme, et de comprendre comment deux enzymes similaires effectuent des fonctions différentes.

Cette compréhension nécessite des expérimentations, des observations manuelles longues et minutieuses ainsi que des comparaisons via les outils de la littérature. Les outils de comparaison en trois dimensions comme DALI, TMalign, MICAN ou ShiNi montrent leurs limites en retournant une bonne similarité entre deux structures dont on connaît de base la ressemblance. La nécessité est ici de détecter non plus des similarités de structure mais des divergences de structure.

Actuellement nous ne connaissons aucun outil pour faciliter les analyses manuelles des structures malgré la fréquence quotidienne des analyses de ce type par les chercheurs.

Afin de pallier ce problème et pour accélérer la détection de divergences entre structures similaires, nous avons développé Daijinushi, l'écuyer, un outil d'aide à la détection et l'analyse de dissimilarités entre structures fortement similaires. La méthode diffère de celles précédemment évoquée car elle est post optimale, c'est à dire qu'elle se base sur une superposition de structures préexistantes, générée manuellement ou via un outil d'alignement de séquence ou structure. L'optimalité est ici relative à la méthode utilisée. Daijinushis'appuie sur des structures superposées de manière optimale (selon un outil classique) puis recherche toutes les différences au niveau des groupes fonctionnels d'une structure à l'autre. Daijinushiest un outil de comparaison deux à deux étendu avec une version permettant de traiter les résultats pour un ensemble de protéines face à une protéine cible et nous avons également un module d'extension pour favoriser la visualisation avec l'outil UCSF Chimera.

Pour présenter cet outil, nous avons tout d'abord comparé deux GH de la famille 16 (**1UMZ,A** et **2UWC,A**) puis nous avons utilisé un ensemble de 11 protéines issues de la

famille GH5, sous-famille 4. Ces protéines similaires structurellement mais séquentiellement assez éloignées nous ont permis d'observer la similarité dite fonctionnelle, c'est-à-dire au niveau des $FGS$ entre une paire de structures données.

## 9.2 Approche par l'exemple du problème

La comparaison des séquences de deux protéines trouve ses limites lorsqu'au cours de l'évolution celles-ci divergent jusqu'à n'être identiques qu'à un faible pourcentage. Par exemple, les protéines **1UMZ,A** et **2UWC,A** (cf figure 9.1) ont une identité de séquence (calculée avec blastp) de 39%. Ces deux enzymes sont une xyloglucan endotransglycosylase (**1UMZ,A**) et une xyloglucane hydrolase (**2UWC,A**). Elles sont annotées du même numéro EC : 2.4.1.207 (selon leurs entrées pdb au 09/09/2015), c'est à dire qu'elles sont toutes deux classifiées du type glycosyltransférase (2.4.x.x). Ce sont deux glycosides hydrolases (famille 16) qui présentent l'intérêt d'avoir deux structures similaires mais une fonction différente. Cela alors que la structure est bien conservée avec 240 paires de résidus alignés (pour 267 et 266 résidus au sein des protéines respectivement) et un RMSDc égal à 2.067Å. En terme de similarité, cela correspond à 90% de similarité de structure.

$$s_{sum} = \frac{2 \times N_e}{N_1 + N_2} = \frac{2 \times 240}{267 + 266} = 0.901$$

FIGURE 9.1 – Superposition des protéines 1UMZ,A et 2UWC,A, MICAN retourne une superposition avec une très faible déviation, ce qui signifie que les structures sont très proches géométriquement

Ces deux protéines sont donc structurellement très similaires, le paradigme structure-fonction suppose qu'elles ont des fonctions proches. Et en effet, leurs fonctions sont en quelque sorte proches puisque ces deux enzymes ont le même substrat (du xyloglucane) et la différence se trouve dans l'action, une enzyme lysant le substrat, l'autre le liant.

En nous basant sur ce constat nous avons cherché à affiner l'étude de ces structures en les observant non plus selon leurs squelettes mais selon la position de leurs groupes fonctionnels

après superposition. Nous ne cherchons plus ici à détecter la similarité de structures mais la divergence entre structures similaires. Nous avons donc développer un outil et un protocole permettant de cibler les groupes fonctionnels divergents d'une structure à l'autre.

## 9.3 Méthodologie

Le protocole est constitué de deux étapes majeures : une superposition des structures et une analyse des groupes fonctionnels.

### 9.3.1 Superposition des structures

La superposition des structures s'effectue avec l'heuristique MICAN. La question d'utiliser ShiNi avec le mode $FGS$ (soit directement avec les groupes fonctionnels) ou de manière classique avec le mode "squelette" ($C\alpha$) s'est posée et la comparaison des superpositions entre les modes $C\alpha$, $FGS$et l'outil MICAN a montré que les trois retournaient une superposition similaire. La figure 9.2 montre un exemple de superposition, on remarque que les structures se superposent de la même façon. Pour comparer ces superpositions nous avons recherché le plus long alignement tel que les distances entre paires de résidus alignés soient inférieures à 2.0Å. Les résultats obtenus pour MICAN et ShiNi($FGS$) sont deux alignements pratiquement identiques. Cela tend à montrer que les deux superpositions créées par ces outils sont semblables. Par conséquent, des trois outils nous choisissons le plus rapide tout en prévoyant une analyse comparative des superpositions issues des modes à plus grande échelle.

Cette étape est majeure car toute l'analyse repose sur la position géométrique des groupes fonctionnels après superposition optimale des structures. Donc, l'utilisation d'un autre algorithme de superposition pourrait significativement modifier le résultat de l'étape suivante.

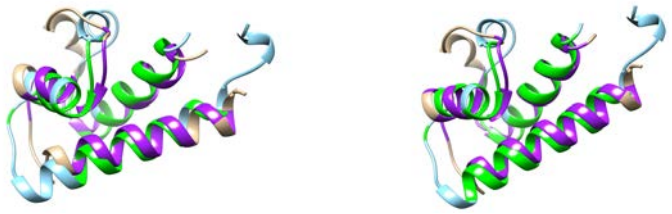


FIGURE 9.2 – Superposition de deux structures (vues du squelette sous forme ribbon) avec MICAN (à g.) et ShiNi mode $FGS$(à d.).
En vert/violet sont représentées les paires alignées dont la distance est inférieure à 2.0Å.

### 9.3.2 Analyse des groupes fonctionnels

A partir des coordonnées transformées des deux structures, nous comparons les deux structures à l'aide de Daijinushi.

Daijinushi(littéralement "l'écuyer") est un outil servant à aider l'utilisateur à cibler des zones, des motifs au sein des structures superposées qui divergent. L'outil commence par modéliser tous les groupes fonctionnels au sein des structures, puis il identifie les groupes qui, d'une structure à l'autre sont :

1. *identiques*
2. *substitués*
3. *spécifiques à l'une ou l'autre des protéines*

**Définition 9.1 ($FGS$ identiques)** *Deux groupes fonctionnels sont dits identiques si, d'une structure à l'autre, il se trouve à la même position (plus ou moins $\iota$Å) un groupe fonctionnel de même nature.*

Le seuil de tolérance utilisé par défaut est $\iota = 2.0$Å. Les $FGS$ identiques sont la partie invariante des structures et ne tiennent pas compte des résidus dont les $FGS$ sont issus. Ainsi les séquences peuvent évoluer, les groupes fonctionnels considérés peuvent également provenir de résidus différents (suite à une coévolution de deux résidus par exemple). Le seul critère d'évaluation des $FGS$ est ici la présence ou non d'un groupe fonctionnel de même nature à la même position au sein des deux structures (après superposition).

**Définition 9.2 ($FGS$ substitués)** *On parle de groupes fonctionnels substitués si, d'une structure à l'autre, il se trouve à la même position (plus ou moins $\iota$Å) un groupe fonctionnel de nature différente.*

Observer ces substitutions et leur localisation devrait permettre de fournir des pistes quant à la variation de fonction entre deux structures, surtout si elles interviennent dans des sites clés des protéines comme les sites catalytiques ou la petite couronne derrière le site.

**Définition 9.3 ($FGS$ spécifiques - indels )** *On parle de groupes fonctionnels spécifiques si un groupe fonctionnel n'est présent que dans l'une des structures.*

Daijinushianalyse les protéines et sépare les différents groupes fonctionnels selon ces catégories. Puis, via une extension créée pour le logiciel de visualisation UCSF Chimera [98] [1], l'utilisateur peut visualiser les zones de divergences. Les groupes fonctionnels pouvant être colorés par nature ou bien par catégorie (identiques, substitués, indels), sur l'exemple suivant (figure 9.3), les groupes fonctionnels de chaque structure sont représentés par une sphère. A chaque $FGS$ est associée une couleur, vert s'il est identique dans l'autre structure, orange si substitution et rouge pour les indels.

Daijinushicalcule également les liaisons hydrogènes et détermine également lesquelles sont maintenues.

1. `http://www.cgl.ucsf.edu/chimera/`

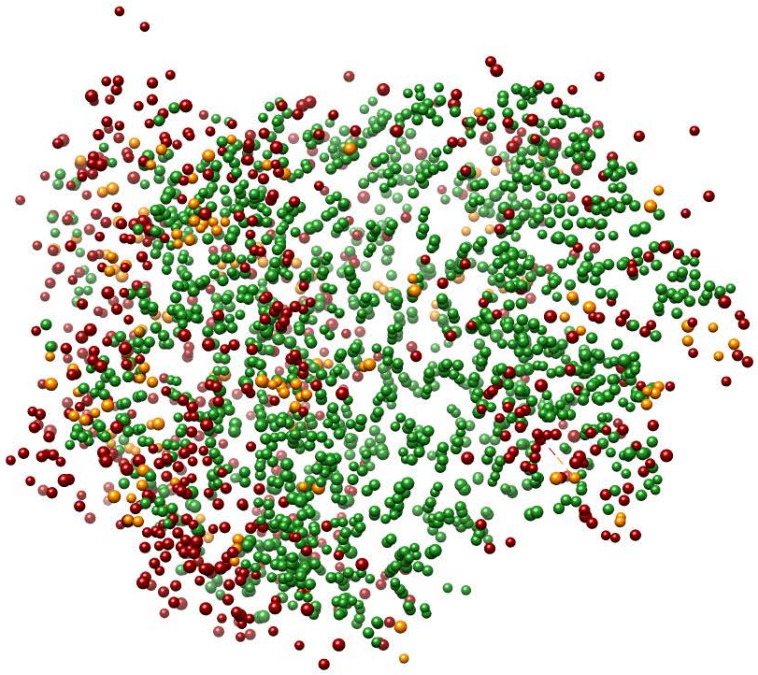


FIGURE 9.3 – Visualisation de l'analyse en groupes fonctionnels des protéines 1UMZ,A et 2UWC,A

### 9.3.3 Résultats

L'objectif de Daijinushiest de permettre à l'utilisateur de visualiser plus facilement la superposition ne lui permettant de sélectionner l'une ou l'autre des catégories. Dans le cas de **1UMZ,A** et **2UWC,A**, l'outil montre deux substitutions dans la gorge et des groupes fonctionnels à l'extrémité de la gorge qui sont spécifiques à l'une des structures. Ces légères différences pourraient aider à la formulation de l'explication de la différence de fonction entre les deux protéines.

### 9.3.4 Analyse détaillée des structures issues des GH5, sous-famille 4

Le jeu de données employé ici est constitué de 11 structures protéiques issues de la famille GH5 (sous-famille 4). Ce classement suppose une forte similarité entre les protéines néanmoins l'identité de séquence de ces structures est très variable (de moins de 30% à 100% selon la paire de structures analysée). Les tableaux suivants récapitulent le pourcentage d'identité de séquence des structures (tableau 9.1) ainsi que la couverture de l'alignement (tableau 9.2). Le pourcentage d'identité de séquence des paires de résidus alignés varie de 24.2 % (**3ZMR,A** vs **4V2X,A**) à pratiquement 100 % (**3AYR,A** vs **3AYS,A**). Cet écart reflète la divergence entre ces protéines appartenant à la même sous-famille fonctionnelle.

La matrice 9.2 n'est pas symétrique car la couverture (le pourcentage de résidus alignés)

TABLE 9.1 – Pourcentage d'identité de séquence des structures après alignement par MICAN
En bleu les protéines au numéro EC 3.2.1.4 et en rouge les protéines au numéro EC 3.2.1.151

| | 4IM4 | 3AYS | 3AYR | 3NDY | 2JEP | 4V2X | 4NF7 | 3ZMR | 1EDG | 3VDH | 4W85 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 4IM4 | 100 | 44 | 44.1 | 61.3 | 37.7 | 33.8 | 38.6 | 35.9 | 41.2 | 36.5 | 41.8 |
| 3AYS | 44 | 100 | 99.7 | 40.4 | 30.6 | 28.9 | 35.3 | 29.8 | 36.5 | 34.3 | 39.4 |
| 3AYR | 44.1 | 99.7 | 100 | 40.4 | 30.6 | 29.2 | 35.4 | 30.1 | 36.8 | 35.7 | 39.7 |
| 3NDY | 61.3 | 40.4 | 40.4 | 100 | 35.6 | 31.5 | 41 | 31.5 | 41.6 | 34 | 38 |
| 2JEP | 37.7 | 30.6 | 30.6 | 35.6 | 100 | 30.8 | 33.4 | 35.7 | 31 | 29.7 | 31.9 |
| 4V2X | 33.8 | 28.9 | 29.2 | 31.5 | 30.8 | 100 | 29.8 | 24.2 | 28.1 | 28.4 | 30.7 |
| 4NF7 | 38.6 | 35.3 | 35.4 | 41 | 33.4 | 29.8 | 100 | 29 | 45.3 | 30.1 | 35 |
| 3ZMR | 35.9 | 29.8 | 30.1 | 31.5 | 35.7 | 24.2 | 29 | 100 | 29.5 | 30 | 28.3 |
| 1EDG | 41.2 | 36.5 | 36.8 | 41.6 | 31 | 28.1 | 45.3 | 29.5 | 100 | 34 | 37.8 |
| 3VDH | 36.5 | 34.3 | 35.7 | 34 | 29.7 | 28.4 | 30.1 | 30 | 34 | 100 | 35.5 |
| 4W85 | 41.8 | 39.4 | 39.7 | 38 | 31.9 | 30.7 | 35 | 28.3 | 37.8 | 35.5 | 100 |

est dépendant de la structure cible (ligne). Les similarités structurales sont grandement plus élevées, supérieures à 55% et allant jusqu'à 100% de recouvrement. Les alignements obtenus ont des RMSDc variant entre 0.272 et 2.546 (avec 50% des valeurs inférieures à 1.928). Ces alignements ont donc des déviations faibles, les structures sont très similaires étant données leurs grandes couvertures. Nous observons des différences mais non significatives selon le numéro EC de la protéine.

TABLE 9.2 – Couverture optimale de la structure cible (ligne) par la structure requête (colonne) après alignement optimal par MICAN
En bleu les protéines au numéro EC 3.2.1.4 et en rouge les protéines au numéro EC 3.2.1.151

| | 4IM4 | 3AYS | 3AYR | 3NDY | 2JEP | 4V2X | 4NF7 | 3ZMR | 1EDG | 3VDH | 4W85 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 4IM4 | 100 | 98.5 | 98.8 | 99.7 | 96.1 | 95 | 98.5 | 96.7 | 97.9 | 89.3 | 97.3 |
| 3AYS | 93 | 100 | 100 | 93.6 | 92.4 | 90.2 | 94.4 | 92.2 | 94.4 | 88.2 | 93.8 |
| 3AYR | 91 | 97.5 | 100 | 92.1 | 90.2 | 88 | 92.6 | 90.7 | 92.1 | 84.2 | 91.5 |
| 3NDY | 96.3 | 95.7 | 96.6 | 100 | 94.3 | 92 | 95.7 | 93.7 | 95.7 | 86 | 94.3 |
| 2JEP | 82.9 | 84.4 | 84.4 | 84.1 | 100 | 82.1 | 84.1 | 88.7 | 85.9 | 81.1 | 83.4 |
| 4V2X | 59 | 59.4 | 59.4 | 59.2 | 59.2 | 100 | 59.4 | 60.1 | 58.5 | 55.9 | 58.9 |
| 4NF7 | 91.5 | 92.8 | 93.4 | 92 | 90.6 | 88.7 | 100 | 91.2 | 94.2 | 88.7 | 92 |
| 3ZMR | 68.5 | 69.1 | 69.7 | 68.7 | 72.9 | 68.5 | 69.5 | 100 | 69.7 | 65.8 | 67.6 |
| 1EDG | 86.8 | 88.7 | 88.7 | 87.9 | 88.4 | 83.4 | 90 | 87.4 | 100 | 81.3 | 88.4 |
| 3VDH | 89.1 | 93.2 | 91.1 | 88.8 | 93.8 | 89.6 | 95.3 | 92.6 | 91.4 | 100 | 89.9 |
| 4W85 | 96.8 | 98.8 | 98.8 | 97.1 | 96.2 | 94.1 | 98.5 | 95 | 99.1 | 89.7 | 100 |

Les grandes similarités structurales et les faibles similarités de séquences font de ces données un bon jeu de test pour la recherche de divergences entre groupes fonctionnels. A partir des alignements de MICAN, nous avons calculé les coordonnées transformées des structures puis les avons analysées deux à deux avec Daijinushi. Soient deux protéines $P_1$ et

$P_2$ composées réciproquement de $n$ et $m$ $FGS$. $N_i$ est le nombre de groupes fonctionnels identiques, $N_s$ le nombre de substitutions et $N_t$ (resp. $N_q$) le nombre de $FGS$ spécifiques à $P_1$. Nous nous sommes posé la question de la mesure de la similarité fonctionnelle, un score basé sur les $FGS$. Il est possible d'appliquer des scores similaires aux scores de comparaison des squelettes :

$$s_{imin}(P_1, P_2) = \frac{N_i}{min(n,m)} \; ; \; s_{imax}(P_1, P_2) = \frac{N_i}{max(n,m)} \; ; \; s_{isum}(P_1, P_2) = \frac{2 \times N_i}{max(n,m)} \tag{9.1}$$

Ces équations sont les ratio des nombres de groupes fonctionnels identiques par rapport aux nombres de $FGS$ dans les structures. De même les deux équations ci-dessous sont les équivalents pour les $FGS$ substitués et spécifiques à l'une ou l'autre des structures.

$$s_{smin}(P_1, P_2) = \frac{N_s}{min(n,m)} \; ; \; s_{smax}(P_1, P_2) = \frac{N_s}{max(n,m)} \; ; \; s_{ssum}(P_1, P_2) = \frac{2 \times N_s}{n+m} \tag{9.2}$$

$$s_{tmin}(P_1, P_2) = \frac{N_t}{min(n,m)} \; ; \; s_{tmax}(P_1, P_2) = \frac{N_t}{max(n,m)} \; ; \; s_{tsum}(P_1, P_2) = \frac{2 \times N_t}{n+m} \tag{9.3}$$

Le sens de ces équations est assez limité car il n'est pas assez explicite, les structures comparées étant de bases très similaires. Le point faible de ces scores est qu'il restreint la similarité calculée à un type de paire de $FGS$(identique/substituée/spécifique). Il faut ainsi mesurer les trois scores puis les comparer deux à deux avec les scores correspondant à une autre paire de structures pour obtenir un semblant de comparaison. En conclusion ces scores basiques n'ont pas encore été étudiés à une assez grande échelle pour que leur utilité soit prouvée. Si un score dit fonctionnel peut s'avérer utile, il est à prévoir qu'il faille prendre en compte les trois types de paires de $FGS$ et probablement les pondérer dans l'équation. Nous ne sommes donc actuellement pas en mesure de fournir un score de similarité dit fonctionnel.

Cependant, nous souhaitons pouvoir observer les variations entre deux protéines face à une même protéine cible. Nous avons donc choisi des valeurs très simples, liées à la cible, à savoir les pourcentages de $FGS$ identiques/substitués/spécifiques de la structure cible face à des structures requêtes. Les figures 9.4 et 9.5 montrent les différents pourcentages relatifs à **2JEP,A** et **4V2X,A**. On observe pour cette dernière structure un fort taux de $FGS$ spécifiques, cela montre une large portion de la protéine additionnelle par rapport aux autres membres de la famille. Les pourcentages de substitutions varient entre 0 et 11% sur l'ensemble des comparaisons, celui d'identité est compris entre 40 et 98% et la proportion de $FGS$ spécifiques varie entre 1 et 51 %. Ces valeurs dispersées sur une large amplitude alors que l'on se trouve au sein d'une même sous-famille montre que la variabilité fonctionnelle est grande et pourrait donc expliquer les différences de fonction entre deux structures similaires.

### 9.3.5 Etude de cas : 3AYS,A vs 4W85,A

Le but de cet exemple est de montrer une application et de l'approche (via Daijinushi) ainsi que des possibilités concrètes d'étude. Nous avons sélectionné deux structures (**3AYS,A**

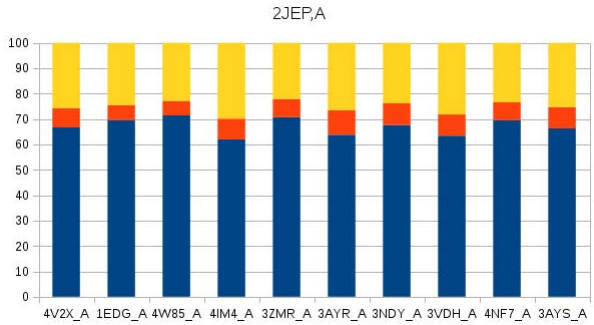



FIGURE 9.4 – Pourcentages de $FGS$ identiques (bleu), substitués (orange) ou spécifiques à 2JEP,A (jaune) par rapport aux 10 autres structures.

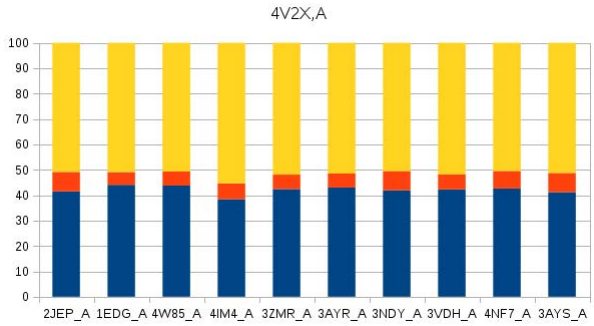



FIGURE 9.5 – Pourcentages de $FGS$ identiques (bleu), substitués (orange) ou spécifiques à 4V2X,A (jaune) par rapport aux 10 autres structures.

et **4W85**), une endoglucanase et une xyloglucanase respectivement. Le fichier contenant la structure **3AYS,A** contient la protéine en complexe avec une molécule : cellotriose. La similarité structurale entre les structures est forte (la couverture de **3AYS**,A est de 93.8% et celle de **4W85** de 98.8%) pour une identité de séquences égale à 39.4% (identité de séquence assez faible). Par conséquent la supposition d'une faible similarité fonctionnelle est envisagée. L'analyse des structures post-superposition montre pourtant une forte similarité fonctionnelle avec 73.83% de $FGS$ identiques, 5.74% de substitutions et respectivement 20.43 et 15.71% de FG spécifiques. Nous avons ensuite placé le ligand (cellotriose)dans les structures conformément au complexe initial. Puis nous avons recherché toutes les modifications proches du site entre les structures (substitutions/indels) chez **3AYS**,A puis **4W85**,A. Pour **3AYS**,A La substitution la plus proche du ligand se trouve à plus de 6.0Å. Sur les 100 substitutions que compte le couple de structures, une quinzaine se trouve dans la couronne proche ($\leq 15$Å$\tilde{)}$. Nous avons ensuite observé les groupes fonctionnels spécifiques à **3AYS**,A étant au voisinage du ligand. Il y en a un à proximité immédiate du ligand ($> 5.0$Å) et les

autres sont au minimum à 7Å de distance. La figure 9.6 montre les $FGS$ au niveau du site de liaison de la cellotriose.

A présent la même figure (fig. 9.7) mais sont représentés cette fois les $FGS$ spécifiques à **4W85**,A (contre **3AYS**,A précédemment). L'analyse des distances montre cette fois des groupes fonctionnels spécifiques proches du ligand, notamment deux accepteurs d'hydrogènes à moins de 2.0Å du ligand et plusieurs autres groupes fonctionnels à moins de 7.0 Å. Ce genre d'observations peut faciliter l'identification et l'explication des divergences de fonctions par les scientifiques.

### 9.3.6 Observations multiples

Daijinushifonctionne à partir d'une paire de protéines superposées. Dans le cadre de l'étude d'une famille, nous avons ajouté une extension qui, à partir d'un ensemble d'analyses de couples de protéines, permet d'observer, à partir d'une référence, les groupes fonctionnels pour un ensemble de protéines.

Pour ce faire, un ensemble de protéines de la famille des GH5 suggéré par l'Equipe Glycobiologie Marine de la station biologique de Roscoff a été utilisée. Nous avons appliqué notre outil sur ces protéines puis sélectionné une protéine "cible" et afficher pour cette protéine toutes les correspondances dans les autres structures. C'est à dire que pour chaque groupe fonctionnel de la protéine cible, l'outil montre s'il existe dans les autres structure un $FGS$ identique, substitué ou rien.

L'extension, figure 9.8, rappelle les outils d'alignements multiples mais ici, toutes les structures sont relatives à la structure cible. N'y est montrée que la présence/absence d'un groupe fonctionnel dans les autres structures.

L'outil liste les groupes fonctionnels de chaque résidu de la protéine cible et, pour l'ensemble des protéines requêtes, récupère la correspondance déterminée par Daijinushi. Les groupes fonctionnels identiques apparaissent en vert et les substitutions en rouge. Un parcours de la structure cible (**1EDG,A**), montre par exemple que dans sa structure, un groupe aromatique porté par une phénylalanine (résidu 25) est substitué dans pratiquement toutes les autres structures (sauf **4NF7**) par un groupement aliphatique. Les cycles aromatiques pouvant se comporter comme des groupements aliphatiques, cette substitution peut n'avoir aucun impact sur la fonction des protéines mais l'outil liste tous les changements et peut donc détecter des différences plus significatives. Autre exemple, le groupe aliphatique de l'alanine du résidu 47 est substitué par un groupe donneur-accepteur d'hydrogène seulement chez **3NDY** et provoque ici un changement de polarité. Donc, cet outil permet de visualiser les motifs invariants entre les structures (correspondant aux sections vertes) mais aussi de détecter les motifs spécifiques à une protéine (ici la structure cible débute par une boucle que n'ont pas les autres protéines d'où les tirets).

## 9.4 Discussion, perspectives

Daijinushi, dans sa version simple (analyse d'un couple de protéines) ou par son extension (analyse pseudo-multiples, pseudo- car tout se réfère à une seule structure) , est un

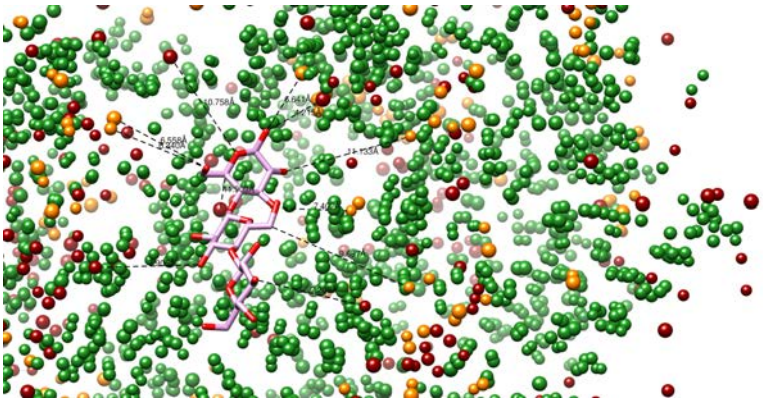


FIGURE 9.6 – Superposition des structures **3AYS**,A et **4W85**,A vue en groupes fonctionnels au niveau du site de liaison de la cellotriose
En rose le ligand, les sphères vertes, oranges, rouges représentent les $FGS$ identiques, substitués et spécifiques à **3AYS**,A respectivement.

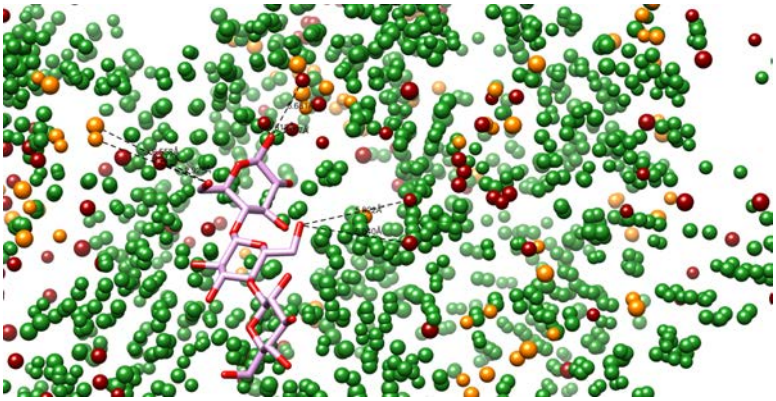
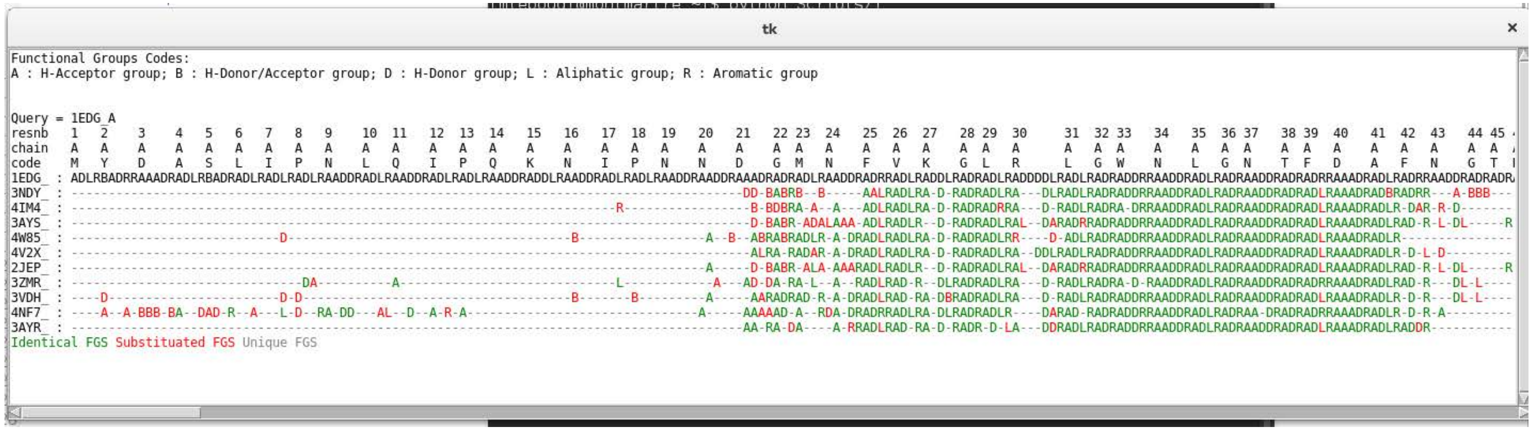


FIGURE 9.7 – Superposition des structures **3AYS**,A et **4W85**,A vue en groupes fonctionnels au niveau du site de liaison de la cellotriose
En rose le ligand, les sphères vertes, oranges, rouges représentent les $FGS$ identiques, substitués et spécifiques à 4W85,A respectivement.



```
Functional Groups Codes:
A : H-Acceptor group; B : H-Donor/Acceptor group; D : H-Donor group; L : Aliphatic group; R : Aromatic group


Query = 1EDG_A
resnb   1   2    3    4   5   6   7   8   9    10  11   12  13  14   15   16   17  18  19   20   21   22 23  24   25  26  27   28 29  30     31  32 33   34   35  36 37   38 39  40   41  42  43   44 45
chain   A   A    A    A   A   A   A   A   A    A   A    A   A   A    A    A    A   A   A    A    A    A  A   A    A   A   A    A  A   A      A   A  A    A    A   A  A    A  A   A    A   A   A    A  A
code    M   Y    D    A   S   L   I   P   N    L   Q    I   P   Q    K    N    I   P   N    N    D    G  M   N    F   V   K    G  L   R      L   G  W    N    L   G  N    T  F   D    A   F   N    G  T
1EDG_ : ADLRBADRRAAADRADLRBADRADLRADLRADLRAADDRADLRAADDRADLRADLRAADDRADDLRAADDRADLRADLRAADDRAADDRAAADRADRADLRAADDRADRRADLRADDLRADRADLRADDDDLRADLRADRADDRRAADDRADLRADRAADDRADRADRRAAADRADLRADRRAADDRADRADR
3NDY_ : ------------------------------------------------------------------------------------------DD-BABRB--B-----AALRADLRA-D-RADRADLRA---DLRADLRADRADDRRAADDRADLRADRAADDRADRADLRAAADRADBRADRR---A-BBB---
4IM4_ : -------------------------------------------------------------------------R-----------------B-BDBRA-A--A---ADLRADLRA-D-RADRADRRA---D-RADLRADRA-DRRAADDRADLRADRAADDRADRADLRAAADRADLR-DAR-R-D-------
3AYS_ : -------------------------------------------------------------------------------------------D-BABR-ADALAAA-ADLRADLR--D-RADRADLRAL--DARADRRADRADDRRAADDRADLRADRAADDRADRADLRAAADRADLRAD-R-L-DL-----R
4W85_ : ----------------------------D--------------------------------------B-----------------A--B--ABRABRADLR-A-DRADLRADLRA-D-RADRADLRR----D-ADLRADRADDRRAADDRADLRADRAADDRADRADLRAAADRADLR---------------
4V2X_ : -------------------------------------------------------------------------------------------ALRA-RADAR-A-DRADLRADLRA-D-RADRADLRA--DDLRADLRADRADDRRAADDRADLRADRAADDRADRADLRAAADRADLR-D-L-D---------
2JEP_ : -------------------------------------------------------------------------------------A-----D-BABR-ALA-AAARADLRADLR--D-RADRADLRAL--DARADRRADRADDRRAADDRADLRADRAADDRADRADLRAAADRADLRAD-R-L-DL-----R
3ZMR_ : -------------------------------DA----------A-----------------------------L------------A---AD-DA-RA-L--A--RADLRAD-R--DLRADRADLRA---D-RADLRADRA-D-RAADDRADLRADRAADDRADRADRRAAADRADLRAD-R---DL-L----
3VDH_ : ----D-----------------------D-D------------------------------------B-------B---------A-----AARADRAD-R-A-DRADLRAD-RA-DBRADRADLRA---D-RADLRADRADDRRAADDRADLRADRAADDRADRADLRAAADRADLR-D-R---DL-L----
4NF7_ : ----A--A-BBB-BA--DAD-R--A---L-D--RA-DD---AL--D--A-R-A-------------------------------A-----AAAAAD-A--RDA-DRADRRADLRA-DLRADRADLR----DARAD-RADRADDRRAADDRADLRADRAA-DRADRADRRAAADRADLR-D-R-A---------
3AYR_ : ------------------------------------------------------------------------------------------AA-RA-DA----A-RRADLRAD-RA-D-RADR-D-LA---DDRADLRADRADDRRAADDRADLRADRAADDRADRADLRAAADRADLRADDR-----------
Identical FGS Substituated FGS Unique FGS
```

FIGURE 9.8 – Visualisation de la comparaison des protéines de la famille GH5 (sur la référence 1EDG)

outil d'aide à l'analyse de structures au niveau des groupes fonctionnels. Il sert à faciliter le parcours du biologiste au sein des structures pour aider à la détection de divergences significatives en élaguant les similarités. La clef de cet outil est son interactivité, il liste toutes les similarités/divergences entre les structures et permet de les retrouver aisément via Chimera. Pour chaque instance (comparaison de deux structures), Daijinushifournit un ensemble de scripts permettant de visualiser les différentes caractéristiques des structures (groupes identiques, substitués...).

Néanmoins il est encore en phase de développement, de nombreuses améliorations sont prévues :

- Une optimisation du regroupement, selon la distance, des groupes fonctionnels à un point donné (centre de la protéine ou site spécifique -catalytique par exemple). Cette fonctionnalité a pour but de distinguer les différentes "couches de la protéine" : le site catalytique, sa petite couronne, ou si le point fourni est le centre de la protéine une différence entre le coeur de la protéine et les couches de surface.
- Regroupement des divergences par analyse des distances entre groupes fonctionnels. L'identification des divergences se fait actuellement selon les groupes fonctionnels pris deux à deux, or il serait intéressant d'identifier des zones plus larges de divergence. Par conséquent nous allons calculer les distances, pour une paire de structures donnée, entre les groupes fonctionnels substitués ou spécifiques. A partir de ces distances et de valeurs de cutoff, nous espérons pouvoir identifier des nuages de groupes fonctionnels correspondant à des zones spécifiques de divergences.
- Une interaction accrue avec l'utilisateur. Daijinushiest actuellement un outil indépendant existant partiellement sous forme d'extension pour le visualiseur Chimera, intégrer toutes les fonctionnalités de l'outil à l'extension serait judicieux pour à la fois faciliter et améliorer le travail de l'utilisateur. En effet l'interactivité accrue permettrait à l'utilisateur de sélectionner rapidement et simplement les informations qu'il recherche. A l'heure actuelle les informations sont disponibles dans les fichiers résultats et une démarche utilisateur est nécessaire pour coupler ces informations avec le visuel de Chimera. L'extension palliera cette faiblesse.

Nous réfléchissons également à des améliorations de l'extension pseudo-multiples en permettant une visualisation des données en trois dimensions afin d'expliciter la réduction 2D des alignements.

Les protéines utilisées étaient des glycosides hydrolases de la famille 5, sous-famille 4, donc définies comme très similaires. Une analyse globale intéressante serait d'observer les zones identiques de cette sous-famille et d'ensuite ajouter une structure plus éloignée (d'une autre sous-famille) pour déterminer les éléments caractéristiques à cette sous-famille. Cette analyse se rapproche des travaux en caractérisation et classification des familles protéiques, mais en ignorant totalement la notion de séquence. Cette abstraction implicite de la séquence est un point fondamental de cette étude car ici le résidu d'origine d'un groupe fonctionnel est totalement occulté, seules sa nature et sa position au sein d'une structure sont prises en compte. Cela induit que les substitutions de résidus n'apparaissent pas directement, donc les événements évolutifs correspondants non plus. Cependant, ces données sont séparées et disponibles, Daijinushirecense toute correspondance de groupes fonctionnels identiques ou

substitués ayant des résidus de nature différente. Ces données présentent un intérêt car elles permettent de montrer qu'une fonction est conservée à une position donnée alors que le gêne codant pour la protéine a subi une mutation.

## 9.5 Conclusion

Daijinushin'est pas un outil de comparaison standard dans le sens où il ne calcule ni alignement, ni score de similarité entre deux structures. Il est une aide à la visualisation de divergences entre structures similaires avec un niveau de précision supérieur à celui des comparateurs de squelettes (centrés sur les $C\alpha$ donc) car il affine la structure via les groupes fonctionnels. L'outil fonctionne à partir des coordonnées transformées par superposition des structures, nous avons choisi de prendre les transformations de MICAN mais n'importe quel autre outil peut être utilisé. L'une des solutions possibles, pour deux structures existantes, est également de créer un alignement des séquences (par BLASTp[111] ou autre) ou des structures, de récupérer l'alignement et d'en calculer la matrice de rotation/translation avec Samourai. A partir de cette matrice et des coordonnées atomiques il est simple de transformer les coordonnées (eg. avec Henge -"transformation"-, notre outil maison) pour ensuite les analyser avec Daijinushi. Cet outil a pour objectif de rajouter, cibler, des informations biologiques aux alignements de structures classiques pour expliciter les cas de divergences de fonction entre structures proches.

Les résultats montrent qu'il est rapidement possible d'identifier des différences intéressantes en se focalisant sur une zone particulière des structures comme un site de liaison. L'observation d'un ensemble de protéines issues d'une sous-famille fonctionnelle a montré un grande variabilité entre les groupes fonctionnels alors que la similarité structurale est importante. Le fait que toutes les analyses se fixent sur une structure de référence ne nous a pas encore permis de déterminer un score qui pourrait être représentatif en une seule valeur de cette similarité fonctionnelle.

En conclusion, la visualisation des structures en 3D via leurs $FGS$ est une piste intéressante d'aide à la compréhension des mécanismes d'actions des protéines (ici des enzymes) et à l'identification et la compréhension des divergences entre structures proches. Nos cas d'études ont montré des pistes d'explications de divergences de fonctions en pointant des éléments différents, chimiquement parlant, entre des structures similaires.

## 9.6 Résumé du chapitre

Nous avons présenté Daijinushi, un outil d'aide à la détection de dissimilarités entre protéines structurellement et fonctionnellement proches. A partir de structures superposées Daijinushirecherche tous les groupes fonctionnels et leurs correspondances (présentes ou non) d'une structure à l'autre. Cela permet de mettre en avant des positions spécifiques des protéines où un changement important s'est effectué (changement de polarité par exemple) sans qu'il y ait de changement dans la structure. Ou à l'inverse cela montre la neutralité de certaines mutations qui modifient la séquence mais qui n'interfèrent pas dans la fonction

de la protéine (le groupe fonctionnel étant maintenu par un autre résidu). Nous avons basé notre outil sur des enzymes, des glycosides hydrolases (des familles 16 d'un côté et 5 de l'autre) et montré que notre outil permettait de cibler des différences entre protéines de même famille. Nous travaillons à l'amélioration de Daijinushiet de ses extensions (dont un module pour le visualiseur Chimera) avec des partenaires afin qu'il puisse être efficace et facilement utilisable par l'ensemble de la communauté. L'interactivité est maître mot de cet outil, "l'écuyer" n'est pas un comparateur de protéines mais un assistant à la compréhension de différences de fonction entre protéines similaires.

# Conclusion générale

## Classification de protéines, comparaison globale de structures

### Contributions

Le nombre de structures ajoutées dans la Protein Data Bank augmente chaque semaine, leur assignation rapide au sein des classifications hiérarchiques est nécessaire et ne peut se permettre d'être uniquement basée sur l'expertise humaine. En créant de nouveaux protocoles basés sur des notions de dominances entre instances, nous avons réussi à conserver une exactitude tout en fournissant des méthodes et outils performants. Nos outils ont montré un taux de réussite supérieur (correcte assignation) à 95% sur des bases de données acceptées par la communauté du domaine (SCOPCATH). De plus nous avons une méthode en cours d'évaluation qui reconnaît les cas où la prédiction est erronée, par conséquent, les domaines mal assignés peuvent donc être écartés de la classification et soumis à d'autres analyses. Nous avons commencé par utiliser un score de similarité entre deux domaines qui se base sur la mesure de recouvrement de cartes de contacts (CMO). Le score CMO (calculé exactement ou d'une manière approchée) permet d'obtenir des mesures de similarités/distances entre protéines. Cette mesure, calculée par un outil conçu et implémenté au sein de l'équipe Genscale, est renvoyée sous forme de bornes qui encadrent la valeur finale. Grâce à ces bornes, il est souvent inutile de résoudre entièrement les comparaisons pour connaître la protéine la plus similaire à une protéine requête parmi un ensemble de protéines donné. Ce procédé se nomme la dominance directe entre instances et nous a permis de significativement réduire la durée nécessaire à l'assignation d'un domaine à une superfamille protéique. Ultérieurement, nous avons prouvé que l'une de nos mesures était une distance métrique ce qui a permis de caractériser l'espace des protéines et par conséquent d'utiliser des propriétés comme l'inégalité triangulaire pour borner la distance entre deux domaines à partir de distances d'autres domaines. Cela, couplé à la dominance (appelée ici dominance indirecte), élague un certain nombre de comparaisons sans qu'aucune résolution, même partielle, du problème de recouvrement de cartes de contacts ne soit effectuée. Ainsi les coûts temps et ressources sont réduits car il n'est plus toujours nécessaire de résoudre ce problème. Ensuite, nous pouvons réduire l'ensemble de l'espace de recherche en éliminant d'emblée des superfamilles grâce à une dominance s'appliquant directement sur les superfamilles de protéines et non sur les domaines protéiques eux-mêmes. Cette procédure permet d'éliminer, d'un bloc, l'ensemble des domaines d'une famille, de la liste des domaines protéiques les plus proches du domaine

requête avec certitude. Enfin, nous avons une mesure, un score normalisé, permettant de vérifier si un domaine protéique est correctement assigné dans la classification ce qui permet de n'intégrer que les domaines correctement prédits et ainsi pallier les erreurs du protocole.

### Quelques perspectives

Nous avons testé une partie des mesures et dominances conçues mais le protocole global nécessite encore des tests avant une mise à disposition. Trois problèmes majeurs se posent :

- la sélection du nombre de domaines structuraux qui entrent en considération pour l'assignation du domaine requête à sa famille
- le pourcentage d'erreur après un protocole mené jusqu'à l'exactitude
- le paramétrage du protocole via une utilisation exacte ou approchée

Selon nos études, lorsque le nombre de domaines participant à l'assignation (kNN) est élevé (entre 5 et 10), la précision de la méthode diminue. Cela tendrait à montrer qu'il vaudrait mieux se restreindre à la recherche du domaine le plus proche. Néanmoins, un vote limite le nombre d'erreurs dues au biais de similarité de structures. Par conséquent nous travaillons à une méthode permettant de prendre en compte les différents domaines intervenant dans le choix de la superfamille mais de manière pondérée et en détectant des erreurs d'assignation. Ces erreurs sont visibles lorsque l'on observe les valeurs des distances entre le domaine requête et les participants au vote. La possibilité de les détecter automatiquement et ainsi d'affiner le protocole est donc à investiguer.

Il reste un certain nombre de mauvaises prédictions, à défaut de pouvoir améliorer la méthode pour les éviter, la capacité de les détecter est un point essentiel. Notre mesure de détection fonctionne sur l'ensemble des données de test, néanmoins il reste à la valider à plus grande échelle. Enfin, nous avons testé notre méthode de manière approchée avec de bons résultats, cela nous incite à paramétrer davantage la méthode afin de réduire le taux d'erreur tout en conservant les bonnes performances de la méthode. Pour cela plusieurs pistes sont envisagées.

- Augmenter la valeur seuil définissant un contact entre résidus ($\mu$), les premières expérimentations à $\mu = 10$Å ont en effet montré une dominance directe simple (requête contre base de données) accrue pour un temps seuil de 2 secondes par comparaison de domaines.
- L'apprentissage est également un point important, utiliser les bornes calculées dans une précédente comparaison va réduire l'espace des solutions de la comparaison courante tout en garantissant son exactitude.

Pour terminer, il est prévu de rendre facilement disponibles nos protocoles par l'intermédiaire d'une page web afin d'en faciliter l'utilisation.

## Comparaison locale de structures, alignements structuraux

### Contributions

L'alignement de deux structures protéiques reste un sujet ouvert, actuellement, nous ne sommes pas capables de statuer sur le meilleur alignement. Derrière l'alignement se trouve la question de la similarité de deux structures, de manière précise. Nous cherchons ici à déterminer la ou les sous-structures communes à deux protéines. Seulement ces protéines sont les résultats d'une histoire évolutive plus ou moins partagée. Ainsi elles ont pu diverger et subir différents événements évolutifs ayant pu amener une divergence relative. Si à l'oeil nu (via un outil de visualisation 3D) cette divergence peut être flagrante, une permutation circulaire par exemple qui modifie la séquence mais pas la structure, elle est souvent un obstacle pour certains outils de comparaison qui n'ont pas été conçus pour la détecter. Nous avons voulu être capables de détecter un maximum de ces événements afin d'avoir une vision la plus globale possible d'une paire de structures.

De même l'un de nos objectifs était d'ajouter des contraintes biochimiques aux modèles pour restreindre les solutions à des ensembles compatibles géométriquement et chimiquement.

Il était donc nécessaire de pouvoir choisir les éléments qui pouvaient s'appareiller d'une structure à l'autre selon des règles précises. La règle de base est la représentation de la protéine par ses résidus, que nous avons enrichi par l'orientation des chaînes latérales. Cela permet de mettre en évidence des différences fines entre deux structures, un piste d'application possible est la détection de coévolutions. Une autre contribution ici est la représentation des protéines non plus par les résidus mais par les fonctions chimiques ($FGS$) qui les composent. Ce modèle, déjà utilisé par [62] et [105], est un niveau intermédiaire entre les résidus et les atomes et permet d'affiner la comparaison sans avoir l'explosion combinatoire liée au grand nombre d'atomes (plus exactement cette explosion est moindre). De plus ce niveau écarte le besoin d'estimer la probabilité de substitution d'un acide aminé par un autre puisqu'ici l'appartenance d'un $FGS$ à un résidu est négligeable. Nous nous intéressons qu'à la substitution d'un $FGS$ par un autre. Le modèle atomique est également disponible et a des perspectives liées à la comparaison non plus des protéines entières mais de leurs surfaces. Le module de modélisation implémenté est Shinobi, il modélise une comparaison, une question biologique, avec le formalisme des graphes et il permet de choisir une recherche globale ou locale de similarités selon la recherche effectuée. En cela il répond aux objectifs d'insertion d'informations et de modularité au sein des comparaisons de structures. Néanmoins, les premiers résultats pour ajouter des informations physico-chimiques au niveau résiduel sont restés infructueux, mettre un critère binaire sur la compatibilité ou non de deux résidus élague beaucoup de paires de résidus qui se retrouvent effectivement dans l'alignement final. Cela, car la comparaison des critères deux à deux et la suppression d'une paire dès que l'un des critères n'est pas respecté est trop contraignant.

Une fois la comparaison de deux structures modélisée par un graphe, la recherche des sous-structures communes débute. Nous voulions non pas la plus grande sous-structure commune mais l'ensemble des sous-structures pertinentes. La pertinence est ici un critère géo-

métrique, les critères biologiques étant imposés dans le graphe. Ces recherches d'alignements structuraux sont effectués par Ninjas qui est un solveur de pseudocliques fonctionnant de paire avec Shinobi. Il extrait du graphe les pseudocliques correspondant à des alignements. Ces alignements structuraux sont les sous-structures similaires recherchées. Ces alignements ont des caractéristiques géométriques garanties par le graphe et le solveur. Lorsque l'étude se fait au niveau local, les différents alignements représentent soient une sous-structure spécifique, typiquement un site de liaison, ou, une fois assemblés, peuvent permettre de montrer une certaine flexibilité. De plus l'alignement est non-séquentiel ce qui permet de détecter les cas de permutation circulaire facilement.

La combinaison de ces deux modules nous a permis de trouver des alignements structuraux en accord avec les alignements issus de la littérature que ce soit dans le cadre de non-séquentialité, de flexibilité ou des alignements structuraux linéaires "classiques". Nos alignements sont soit un long alignement rigide, soit un ensemble de petits alignements rigides chaînés pour modéliser une flexibilité. Le coeur de notre outil est la recherche de zones similaires rigides d'une protéine à l'autre. Cette rigidité se pondère par une valeur seuil que l'utilisateur choisi et est donc adaptable à la comparaison effectuée. La comparaison s'effectue en globalité ou de manière locale, ici cela permet de cibler la recherche.

Les problèmes précédents étaient liés à la comparaison de deux structures mais il est également intéressant de pouvoir observer une seule structure. En effet, des études ont montré que certaines structures étaient constitué d'un petit motif se répétant plusieurs fois. Cet effet est particulièrement visuel comme l'ont montré les exemples. Néanmoins certains motifs échappent à l'oeil et par conséquent nous avons réfléchi à des méthodes permettant de détecter automatiquement la présence de ces motifs répétés. Ces motifs se caractérisant par une longueur similaire en termes de nombre de résidus et par une structure identique ou presque. Nous avons utilisé une méthode de découpage de la protéine en dalles. Ensuite nous avons cherché à trouver les dalles identiques pour reconstruire la protéine. A terme ces méthode seront utilisées avec les outils précédemment cités mais nous les avons testées avec un outil simplifié : Kunoichi. Il a été conçu comme outil préliminaire pour l'identification de répétitions dans les structures protéiques pour effectuer un dallage. Malgré ses limites dues à la linéarité de ses dalles, il a fourni de bons résultats. La recherche de cliques basée sur des dalles, des motifs structuraux non-chevauchants, encapsulée dans un protocole *de novo*, permet d'identifier facilement ces motifs répétés. Il reste des améliorations, principalement au niveau de l'intégration de gaps, le passage à la non-linéarité devrait permettre d'intégrer des motifs structuraux impliquant des changements ponctuels de topologies.

Nos recherches produisent nombre d'alignements et afin de vérifier la pertinence de ces alignements, nous avons souhaité les confronter avec ceux issus d'autres méthodes de comparaison. Cependant, chaque méthode ayant ses objectifs, il est délicat de conclure de la pertinence de l'une ou de l'autre. Pour essayer de mieux comprendre ces différents alignements, nous avons souhaité les comparer sur la base de plusieurs scores. Ces scores ont été implémenté dans Samourai , un petit outil qui mesure de nombreux scores à partir d'un alignement. Il nécessite de connaître les structures protéiques et qu'on lui fournisse un ensemble de paires de résidus. Ces nombreux scores sont autant d'informations sur l'alignement. Ils estiment une similarité de manière différente et avoir ainsi un outil qui les mesure tous permet

de comparer des alignements issus d'outils différents.

Les recherches de similarités entre structures sont un cas courant mais parfois celles-ci ne suffisent pas pour comprendre les deux structures. En particulier si celles-ci se ressemblent énormément, alors ce ne sont plus des similarités qu'il faut rechercher mais des divergences. A l'inverse des autres méthodes, l'alignement, la superposition des structures n'est plus la finalité mais la base de l'analyse. Ensuite, nous recensons toutes les fonctions chimiques, identiques ou non, d'une structure à l'autre pour aider l'utilisateur à identifier celles qui manquent, celles qui ont changé, et ainsi nous souhaitons faciliter la compréhension des différences de fonctions entre protéines proches. Cette recherche de divergences entre structures similaires a été implémentée dans Daijinushi, un assistant.

## Perspectives

L'un des objectifs de cette thèse était de modéliser de différentes manières les structures protéiques afin d'essayer de se rapprocher au plus de la réalité biologique. En effet, actuellement la grande majorité des outils d'alignements structuraux se focalisent sur la structure pure, la ressemblance géométrique. Or, la fonction d'une protéine, le point d'orgue de toutes les analyses, n'est pas uniquement dépendante de la structure. Pourtant à l'heure actuelle rares sont les outils mixtes. Historiquement les outils de comparaisons de protéines sont basés sur la séquence, puis sur la structure. Les deux méthodes ont largement fait leurs preuves mais il est aujourd'hui nécessaire de poursuivre vers des modèles mixtes.

La majorité des structures similaires restent linéaires, l'ordre est conservé, les points de flexibilité sont rares, par conséquent une heuristique classique comme TMalign reste une référence. Néanmoins de plus en plus de cas non-linéaires sont découverts et nécessitent des outils adaptés. Actuellement, aucun outil ne permet de détecter tous les cas, et aucun protocole automatique testant ces différents cas n'existe. De fait une automatisation semble une perspective naturelle. Un gros point faible actuel est la multiplicité des outils dont la majorité ne peut être réellement comparée car chacun a un objectif différent même si le but global est d'aligner au mieux les structures. On retrouve donc de plus en plus d'études consensuelles [44][110] qui cherchent à s'affranchir de ces contraintes. Une analyse exhaustive de ces outils et des alignements qu'ils produisent est nécessaire, notamment car nos études ont montré que certains outils convergeaient, ils découvrent la même similarité mais, par des fonctions de scores différentes, l'expriment par des alignements différents. Ensuite, l'étude des protéines, la recherche de leur fonction, la détection des interactions avec d'autres molécules sont de plus en plus étudiées au niveau local, en surface des protéines, à un niveau non plus résiduel mais gros grain ou atomique. L'un des cas que nous n'avons que peu abordé est la détection de résidus co-évoluant, c'est à dire des mutations conjointes qui permettent de maintenir la structure et la présence de fonction chimiques. Quelques tests ont montré que ShinobiNinjas en détectaient mais il est nécessaire de pousser dans cette direction. Enfin concernant les études de surface, le modèle en groupes fonctionnels ($FGS$) est un point de base intéressant car il est ici possible de comparer les structures à un niveau relativement fin tout en ayant une certaine flexibilité et donc de pouvoir détecter des zones similaires non séquentielles. Le modèle de graphe devrait aussi permettre de basculer dans la recherche de zone

de complémentarités : la complémentarité chimique est une simple option déjà implémentée dans Shinobi, la complémentarité géométrique est un point à étudier. De nombreux travaux existent déjà en docking mais dans notre modèle il serait également pertinent d'aller explorer la connaissance au niveau des recherches en modélisation d'objets 3D, animation 3D et jeux vidéos.

Les pistes envisagées pour faciliter la recherche de divergences sont l'utilisation de points "centraux" servant à répartir les fonctions chimiques suivant leurs éloignement par rapport à ces points. De même une sélection plus active des fonctions selon qu'elles soient situées en surface ou au coeur de la protéine constitue des travaux en cours. Enfin, l'interactivité est très importante, il ne s'agit pas ici de mesure mais bien de visualisation, de travail en temps réel avec le chercheur par conséquent Daijinushi est une extension pour le logiciel Chimera en cours de développement.

## Conclusion

Cette thèse fut certes l'occasion de créer des outils et des protocoles d'études des protéines, de leurs structures. Elle servit à modéliser différentes questions biologiques au sein de modèles de graphes. Mais elle a surtout mis en avant la multiplicité des challenges et des différentes réponses déjà existantes. L'étude globale de deux protéines, la mesure de leur similarité et ensuite la difficulté d'assigner correctement un domaine à une famille structural a permis de réfléchir au problème de la synthèse de toutes les similitudes et divergences en une seule et unique valeur : le score. Ce score se doit à lui seul de suffire pour permettre une classification fiable et pertinente, à l'heure actuelle il n'en existe pas de tel. De même la comparaison fine de structures, la recherche de l'alignement optimal pour permettre la compréhension de la protéine est encore un sujet ouvert. Cette thèse pose les questions de la pertinence des alignements, des besoins de les comparer et de la captation des similarités, des éléments communs des protéines à travers ces alignements. Certains outils paraissent fournir des résultats différents mais vu sous un autre angle montrent la même chose et réussir à comprendre pourquoi reste l'un des enjeux de la bioinformatique structurale actuelle.

# Liste de Publications

## Journaux

1. Rumen Andonov, Hristo Djidjev, Gunnar Klau, Mathilde Le Boudic-Jamin, and Inken Wohlers. Automatic Classification of Protein Structure Using the Maximum Contact Map Overlap Metric. Special issue of Algorithms, 2015.
2. Guillaume Chapuis, Mathilde Le Boudic-Jamin, Rumen Andonov, Hristo Djidjev, and Dominique Lavenier. Parallel seed-based approach to multiple protein structure similarities detection. Scientific Programming ,2015

## Conférences internationales

1. Guillaume Chapuis, Mathilde Le Boudic-Jamin, Rumen Andonov, Hristo Djidjev, and Dominique Lavenier. Parallel seed-based approach to protein structure similarity detection. In Parallel Processing and Applied Mathematics, pages 278–287. Springer, 2014.
2. Noel Malod-Dognin, Mathilde Le Boudic-Jamin, Pritish Kamath, and Rumen Andonov. Using dominances for solving the protein family identification problem. In Algorithms in Bioinformatics, pages 201–212. Springer, 2011.
3. Inken Wohlers, Mathilde Le Boudic-Jamin, Hristo Djidjev, GunnarW. Klau, and Rumen Andonov. Exact Protein Structure Classification Using the Maximum Contact Map Overlap Metric. In Adrian-Horia Dediu, Carlos Martín-Vide, and Bianca Truthe, editors, Algorithms for Computational Biology, volume 8542 of Lecture Notes in Computer Science, pages 262–273. Springer International Publishing, 2014.

## Conférences nationales

1. Mathilde Le Boudic-Jamin and Rumen Andonov. Détection de novo de structures répétées au sein des protéines. In JOBIM 2015, Clermont-Ferrand, France, 2015. JOBIM, Université d'Auvergne.
2. Mathilde Le Boudic-Jamin, Noël Malod-Dognin, Alexandre Cornu, Jacques Nicolas, and Rumen Andonov. Identification rapide de familles protéiques par dominance. In 12th Annual Congress of the French National Society of Operations Research and Decision Science (ROADEF), volume 2, pages 791–792, Saint-étienne, France, 2011. école Nationale Supérieure des Mines de Saint-étienne.
3. Mathilde Le Boudic-Jamin, Antonio Mucherino, and Rumen Andonov. Modeling protein flexibility by distance geometry. In ROADEF 2012, Angers, France, 2012. ROADEF, Université d'Angers.

# Bibliographie


[1] Anne-Laure Abraham, Eduardo P C Rocha, and Joël Pothier. Swelfe : a detector of internal repeats in sequences and structures. *Bioinformatics (Oxford, England)*, 24(13) :1536–7, July 2008.

[2] Alexej Abyzov and Valentin A Ilyin. A comprehensive analysis of non-sequential alignments between all protein structures. *BMC structural biology*, 7 :78, 2007.

[3] N N Alexandrov and D Fischer. Analysis of topological and nontopological structural similarities in the PDB : new examples with old structures. *Proteins*, 25(3) :354–65, July 1996.

[4] R Andonov, N Yanev, and K A Crandall. A_purva : User Manual 1. (0) :0–2, 2011.

[5] Rumen Andonov, Hristo Djidjev, Gunnar Klau, Mathilde Le Boudic-Jamin, and Inken Wohlers. Automatic Classification of Protein Structure Using the Maximum Contact Map Overlap Metric. *Algorithms*, 2015.

[6] M a Andrade, C Perez-Iratxeta, and C P Ponting. Protein repeats : structures, functions, and evolution. *Journal of structural biology*, 134(2-3) :117–31, 2001.

[7] Antonina Andreeva, Dave Howorth, Cyrus Chothia, Eugene Kulesha, and Alexey G Murzin. SCOP2 prototype : a new approach to protein structure mining. *Nucleic acids research*, 42(1) :D310–4, January 2014.

[8] Antonina Andreeva, Andreas Prlić, Tim J P Hubbard, and Alexey G Murzin. SISYPHUS–structural alignments for proteins with non-trivial relationships. *Nucleic acids research*, 35(Database issue) :D253–9, January 2007.

[9] Stefano Angaran, Mary Ellen Bock, Claudio Garutti, and Concettina Guerra. MolLoc : a web tool for the local structural alignment of molecular surfaces. *Nucleic acids research*, 37(Web Server issue) :W565–70, July 2009.

[10] Rolf Apweiler, Amos Bairoch, Lydie Bougueleret, Severine Altairac, Valeria Amendolia, Andrea Auchincloss, Ghislaine Argoud-Puy, Kristian Axelsen, Delphine Baratin, Marie Claude Blatter, Brigitte Boeckmann, Jerven Bolleman, Laurent Bollondi, Emmaunuel Boutet, Silvia Braconi Quintaje, Lionel Breuza, Alan Bridge, Edouard DeCastro, Luciane Ciapina, Danielle Coral, Elisabeth Coudert, Isabelle Cusin, Gwennaelle Delbard, Dolnide Dornevil, Paula Duek Roggli, Severine Duvaud, Anne Estreicher, Livia Famiglietti, Marc Feuermann, Sebastain Gehant, Nathalie Farriol-Mathis, Serenella Serenella Ferro, Elisabeth Gasteiger, Alain Gateau, Vivienne Gerritsen, Arnaud

Gos, Nadine Gruaz-Gumowski, Ursula Hinz, Chantal Hulo, Nicolas Hulo, Janet James, Silvia Jimenez, Florence Jungo, Vivien Junker, Thomas Kappler, Guillaume Keller, Corinne Lachaize, Lydie Lane-Guermonprez, Petra Langendijk-Genevaux, Vicente Lara, Philippe Lemercier, Virginie Le Saux, Damien Lieberherr, Tania de Oliveira Lima, Veronique Mangold, Xavier Martin, Patrick Masson, Karine Michoud, Madelaine Moinat, Anne Morgat, Anais Mottaz, Salvo Paesano, Ivo Pedruzzi, Isabelle Phan, Sandrine Pilbout, Violaine Pillet, Sylvain Poux, Monica Pozzato, Nicole Redaschi, Sorogini Reynaud, Catherine Rivoire, Bernd Roechert, Michel Schneider, Christian Sigrist, Karin Sonesson, Sylvie Staehli, Andre Stutz, Shyamala Sundaram, Micheal Tognolli, Laure Verbregue, Anne Lise Veuthey, Lina Yip, Luiz Luiz Zuletta, Rolf Apweiler, Yasmin Alam-Faruque, Ricardo Antunes, Daniel Barrell, David Binns, Lawrence Bower, Paul Browne, Chan Wei Mun, Emily Dimmer, Ruth Eberhardt, Alexander Fedotov, Rebecca Foulger, John Garavelli, Renato Golin, Alan Horne, Rachael Huntley, Julius Jacobsen, Michael Kleen, Paul Kersey, Kati Laiho, Rasko Leinonen, Duncan Legge, Quan Lin, Michele Magrane, Maria Jesus Martin, Claire O'Donovan, Sandra Orchard, John O'Rourke, Samuel Patient, Manuela Pruess, Andrey Sitnov, Eleanor Stanley, Matt Corbett, Giuseppe di Martino, Mike Donnelly, Jie Luo, Peter van Rensburg, Cathy Wu, Cecilia Arighi, Leslie Arminski, Winona Barker, Yongxin Chen, Zhang Zhi Hu, Hsing Kuo Hua, Hongzhan Huang, Raja Mazumder, Peter McGarvey, Darren A. Natale, Anastasia Nikolskaya, Natalia Petrova, Baris E. Suzek, Sona Vasudevan, C. R. Vinayaka, Lai Su Yeh, and Jian Zhang. The Universal Protein resource (UniProt) 2009. *Nucleic Acids Research*, 37(SUPPL. 1), 2009.

[11] Rolf Apweiler, Amos Bairoch, Cathy H Wu, Winona C Barker, Brigitte Boeckmann, Serenella Ferro, Elisabeth Gasteiger, Hongzhan Huang, Rodrigo Lopez, Michele Magrane, Maria J Martin, Darren A Natale, Claire O'Donovan, Nicole Redaschi, and Lai-Su L Yeh. UniProt : the Universal Protein knowledgebase. *Nucleic acids research*, 32(Database issue) :D115–D119, 2004.

[12] Zeyar Aung and Kian-Lee Tan. MatAlign : precise protein structure comparison by matrix alignment. *Journal of bioinformatics and computational biology*, 4(6) :1197–1216, 2006.

[13] a Bahr, J D Thompson, J C Thierry, and O Poch. BAliBASE (Benchmark Alignment dataBASE) : enhancements for repeats, transmembrane sequences and circular permutations. *Nucleic acids research*, 29(1) :323–6, January 2001.

[14] H M Berman, J Westbrook, Z Feng, G Gilliland, T N Bhat, H Weissig, I N Shindyalov, and P E Bourne. The Protein Data Bank. *Nucleic acids research*, 28 :235–242, 2000.

[15] Åsa K. Björklund, Diana Ekman, and Arne Elofsson. Expansion of protein domain repeats. *PLoS Computational Biology*, 2 :0959–0970, 2006.

[16] Åsa K. Björklund, Diana Ekman, Sara Light, Johannes Frey-Skött, and Arne Elofsson. Domain rearrangements in protein evolution. *Journal of Molecular Biology*, 353(4) :911–923, 2005.

[17] S. E. Bliven, P. E. Bourne, and A. Prli. Detection of circular permutations within protein structures using CE-CP. *Bioinformatics*, 31(8) :1316–1318, December 2014.

[18] Spencer Bliven and Andreas Prlić. Circular permutation in proteins. *PLoS computational biology*, 8(3) :e1002445, January 2012.

[19] S E Brenner, P Koehl, and M Levitt. The ASTRAL compendium for protein structure and sequence analysis. *Nucleic acids research*, 28(1) :254–6, January 2000.

[20] Catherine Bru, Emmanuel Courcelle, Sébastien Carrère, Yoann Beausse, Sandrine Dalmar, and Daniel Kahn. The ProDom database of protein domain families : More emphasis on 3D. *Nucleic Acids Research*, 33(DATABASE ISS.), 2005.

[21] Daniel R Caffrey, Shyamal Somaroo, Jason D Hughes, Julian Mintseris, and Enoch S Huang. Are protein –protein interfaces more conserved in sequence than the rest of the protein surface ? pages 190–202, 2004.

[22] Alberto Caprara, Robert Carr, Sorin Istrail, Giuseppe Lancia, and Brian Walenz. 1001 optimal PDB structure alignments : integer programming methods for finding the maximum contact map overlap. *Journal of computational biology : a journal of computational molecular cell biology*, 11(1) :27–52, January 2004.

[23] Nejc Carl, Janez Konc, Blaz Vehar, and Dusanka Janezic. Protein-protein binding site prediction by local structural alignment. *Journal of chemical information and modeling*, 50(10) :1906–13, October 2010.

[24] F Cazals, H Kanhere, and S Loriot. Computing the Volume of a Union of Balls : a Certified Algorithm. pages 1–19, 2011.

[25] Broto Chakrabarty and Nita Parekh. PRIGSA : protein repeat identification by graph spectral analysis. *Journal of bioinformatics and computational biology*, 12(6) :1442009, December 2014.

[26] Guillaume Chapuis, Mathilde Le Boudic-Jamin, Rumen Andonov, Hristo Djidjev, and Dominique Lavenier. Parallel seed-based approach to protein structure similarity detection. In *Parallel Processing and Applied Mathematics*, pages 278–287. Springer, 2014.

[27] Guillaume Chapuis, Mathilde Le Boudic-Jamin, Rumen Andonov, Hristo Djidjev, and Dominique Lavenier. Parallel seed-based approach to multiple protein structure similarities detection. *Scientific Programming*, 2015, 2015.

[28] Luonan Chen, Ling-Yun Wu, Yong Wang, Shihua Zhang, and Xiang-Sun Zhang. Revealing divergent evolution, identifying circular permutations and detecting active-sites by protein structure comparison. *BMC structural biology*, 6(1) :18, January 2006.

[29] Luonan Chen, Ling-Yun Wu, Yong Wang, Shihua Zhang, and Xiang-Sun Zhang. Revealing divergent evolution, identifying circular permutations and detecting active-sites by protein structure comparison. *BMC structural biology*, 6 :18, January 2006.

[30] Hua Cheng, Bong-Hyun Kim, and Nick V Grishin. MALIDUP : a database of manually constructed structure alignments for duplicated domain pairs. *Proteins*, 70(4) :1162–6, March 2008.

[31] Gergely Csaba, Fabian Birzele, and Ralf Zimmer. Systematic comparison of SCOP and CATH : a new gold standard for protein structure analysis. *BMC structural biology*, 9 :23, January 2009.

[32] B. A. Cunningham, J. J. Hemperly, T. P. Hopp, and G. M. Edelman. Favin versus concanavalin A : Circularly permuted amino acid sequences. *Proceedings of the National Academy of Sciences*, 76(7) :3218–3222, July 1979.

[33] Paweł Daniluk and Bogdan Lesyng. A novel method to compare protein structures using local descriptors. *BMC bioinformatics*, 12 :344, January 2011.

[34] O. Dror, H. Benyamini, R. Nussinov, and H. Wolfson. MASS : multiple structural alignment by secondary structures. *Bioinformatics*, 19(Suppl 1) :i95–i104, July 2003.

[35] Ugur Emekli, Dina Schneidman-Duhovny, Haim J Wolfson, Ruth Nussinov, and Turkan Haliloglu. HingeProt : automated prediction of hinges in protein structures. *Proteins*, 70(4) :1219–27, March 2008.

[36] Richard A George and Jaap Heringa. The REPRO server : finding protein internal sequence repeats through the Web. *Trends in biochemical sciences*, 25(10) :515–517, 2000.

[37] Mark Gerstein, Arthur M. Lesk, and Cyrus Chothia. Structural Mechanisms for Domain Movements in Proteins. *Biochemistry*, 33(22) :6739–6749, June 1994.

[38] a Godzik. The structural alignment between two proteins : is there a unique answer ? *Protein science : a publication of the Protein Society*, 5(7) :1325–38, July 1996.

[39] A Godzik, J Skolnick, and A Kolinski. Regularities in interaction patterns of globular proteins. *Protein engineering*, 6(8) :801–810, 1993.

[40] D. Goldman, S. Istrail, and C.H. Papadimitriou. Algorithmic aspects of protein structure similarity. *40th Annual Symposium on Foundations of Computer Science (Cat. No.99CB37039)*, 1999.

[41] Lesley H Greene, Tony E Lewis, Sarah Addou, Alison Cuff, Tim Dallman, Mark Dibley, Oliver Redfern, Frances Pearl, Rekha Nambudiry, Adam Reid, Ian Sillitoe, Corin Yeats, Janet M Thornton, and Christine A Orengo. The CATH domain structure database : new protocols and classification levels give a more comprehensive resource for exploring evolution. *Nucleic acids research*, 35(Database issue) :D291–7, January 2007.

[42] Aysam Guerler and Ernst-Walter Knapp. Strategies of non-sequential protein structure alignments. *Genome informatics. International Conference on Genome Informatics*, 22 :21–29, 2010.

[43] Hitomi Hasegawa and Liisa Holm. Advances and pitfalls of protein structural alignment. *Current opinion in structural biology*, 19(3) :341–8, June 2009.

[44] Jim Havrilla and Ahmet Sacan. Meta-analysis of protein structural alignment. In *2012 IEEE International Conference on Bioinformatics and Biomedicine Workshops*, pages 72–76. IEEE, October 2012.

[45] Matthias Heinig and Dmitrij Frishman. STRIDE : a web server for secondary structure assignment from known atomic coordinates of proteins. *Nucleic acids research*, 32(Web Server issue) :W500–2, July 2004.

[46] B Henrissat. A classification of glycosyl hydrolases based on amino acid sequence similarities. *The Biochemical journal*, 280 ( Pt 2 :309–16, December 1991.

[47] B Henrissat and A Bairoch. New families in the classification of glycosyl hydrolases based on amino acid sequence similarities. *The Biochemical journal*, 293 ( Pt 3 :781–8, August 1993.

[48] L. Holm and J. Park. DaliLite workbench for protein structure comparison. *Bioinformatics*, 16(6) :566–567, June 2000.

[49] L Holm and C Sander. Protein structure comparison by alignment of distance matrices. *Journal of molecular biology*, 233(1) :123–38, September 1993.

[50] L Holm and C Sander. Parser for protein folding units. *Proteins*, 19(3) :256–68, July 1994.

[51] Liisa Holm and Chris Sander. Dali : a network tool for protein structure comparison. *Trends in Biochemical Sciences*, 20(11) :478–480, November 1995.

[52] Lisa Holm, S. Kääriäinen, P. Rosenström, and A. Schenkel. Searching protein structure databases with DaliLite v.3. *Bioinformatics*, 24(23) :2780–2781, 2008.

[53] Thomas Hrabe and Adam Godzik. ConSole : using modularity of Contact maps to locate Solenoid domains in protein structures. *BMC bioinformatics*, 15(1) :119, 2014.

[54] Nicolas Hulo, Amos Bairoch, Virginie Bulliard, Lorenzo Cerutti, Edouard De Castro, Petra S Langendijk-Genevaux, Marco Pagni, and Christian J A Sigrist. The PROSITE database. *Nucleic acids research*, 34(Database issue) :D227–30, January 2006.

[55] Sarah Hunter, Philip Jones, Alex Mitchell, Rolf Apweiler, Teresa K Attwood, Alex Bateman, Thomas Bernard, David Binns, Peer Bork, Sarah Burge, Edouard de Castro, Penny Coggill, Matthew Corbett, Ujjwal Das, Louise Daugherty, Lauranne Duquenne, Robert D Finn, Matthew Fraser, Julian Gough, Daniel Haft, Nicolas Hulo, Daniel Kahn, Elizabeth Kelly, Ivica Letunic, David Lonsdale, Rodrigo Lopez, Martin Madera, John Maslen, Craig McAnulla, Jennifer McDowall, Conor McMenamin, Huaiyu Mi, Prudence Mutowo-Muellenet, Nicola Mulder, Darren Natale, Christine Orengo, Sebastien Pesseat, Marco Punta, Antony F Quinn, Catherine Rivoire, Amaia Sangrador-Vegas, Jeremy D Selengut, Christian J a Sigrist, Maxim Scheremetjew, John Tate, Manjulapramila Thimmajanarthanan, Paul D Thomas, Cathy H Wu, Corin Yeats, and Siew-Yit Yong. InterPro in 2011 : new developments in the family and domain prediction database. *Nucleic acids research*, 40(Database issue) :D306–12, January 2012.

[56] Julien Jorda and Andrey V Kajava. T-REKS : identification of Tandem REpeats in sequences with a K-meanS based algorithm. *Bioinformatics (Oxford, England)*, 25(20) :2632–8, October 2009.

[57] W Kabsch and C Sander. Dictionary of protein secondary structure : pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12) :2577–637, December 1983.

[58] Hua-Ying Kao, Wen-Shyong Tzou, Yen-Chu Hsu, Chien-Ming Chen, and Tun-Wen Pai. IRIS : Internal Repeat Identification System. 2009.

[59] M Karplus, T Ichiye, and B M Pettitt. Configurational entropy of native proteins. *Biophysical journal*, 52(6) :1083–5, December 1987.

[60] Bostjan Kobe and Andrey V Kajava. When protein folding is simplified to protein coiling : the continuum of solenoid protein structures. *Trends in Biochemical Sciences*, 25(10) :509–515, October 2000.

[61] Rachel Kolodny, Patrice Koehl, and Michael Levitt. Comprehensive evaluation of protein structure alignment methods : scoring by geometric measures. *Journal of molecular biology*, 346(4) :1173–88, March 2005.

[62] Janez Konc. An improved branch and bound algorithm for the maximum clique problem. 58 :569–590, 2007.

[63] Janez Konc. Original paper. 26(9) :1160–1168, 2010.

[64] Janez Konc and Dusanka Janezic. ProBiS algorithm for detection of structurally similar protein binding sites by local structural alignment. *Bioinformatics (Oxford, England)*, 26(9) :1160–8, May 2010.

[65] E. Krissinel and K. Henrick. Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions. *Acta Crystallographica Section D : Biological Crystallography*, 60 :2256–2268, 2004.

[66] Evgeny Krissinel. On the relationship between sequence and structure similarities in proteomics. *Bioinformatics (Oxford, England)*, 23(6) :717–23, March 2007.

[67] Giuseppe Lancia, Robert Carr, Brian Walenz, and Sorin Istrail. 101 Optimal PDB Structure Alignments : a Branch–and–Cut Algorithm For The Maximum Contact Map Overlap Problem. pages 193–202, 2001.

[68] Roman a. Laskowski, Victor V. Chistyakov, and Janet M. Thornton. PDBsum more : New summaries and analyses of the known 3D structures of proteins and nucleic acids. *Nucleic Acids Research*, 33(DATABASE ISS.) :266–268, 2005.

[69] R H Lathrop. The protein threading problem with sequence amino acid interaction preferences is NP-complete. *Protein engineering*, 7(9) :1059–68, September 1994.

[70] Mathilde Le Boudic-Jamin and Rumen Andonov. Détection de novo de structures répétées au sein des protéines. In *JOBIM 2015*, Clermont-Ferrand, France, 2015. JOBIM, Université d'Auvergne.

[71] Michael Levitt. Nature of the protein universe. *Proceedings of the National Academy of Sciences of the United States of America*, 106(27) :11079–84, July 2009.

[72] Ylva Lindqvist and Gunter Schneider. Circular permutations of natural protein sequences : structural evidence. *Current Opinion in Structural Biology*, 7(3) :422–427, June 1997.

[73] Loredana Lo Conte, Steven E Brenner, Tim J P Hubbard, Cyrus Chothia, and Alexey G Murzin. SCOP database in 2002 : refinements accommodate structural genomics. *Nucleic acids research*, 30(1) :264–267, 2002.

[74] Noel Malod-dognin, Mathilde Le Boudic-jamin, Pritish Kamath, and Rumen Andonov. Using Dominances for Solving the Protein Family Identification Problem. pages 201–212, 2011.

[75] Aron Marchler-Bauer, Shennan Lu, John B Anderson, Farideh Chitsaz, Myra K Derbyshire, Carol DeWeese-Scott, Jessica H Fong, Lewis Y Geer, Renata C Geer, Noreen R Gonzales, Marc Gwadz, David I Hurwitz, John D Jackson, Zhaoxi Ke, Christopher J Lanczycki, Fu Lu, Gabriele H Marchler, Mikhail Mullokandov, Marina V Omelchenko, Cynthia L Robertson, James S Song, Narmada Thanki, Roxanne A Yamashita, Dachuan Zhang, Naigong Zhang, Chanjuan Zheng, and Stephen H Bryant. CDD : a Conserved Domain Database for the functional annotation of proteins. *Nucleic acids research*, 39(Database issue) :D225–9, January 2011.

[76] E M Marcotte, M Pellegrini, T O Yeates, and D Eisenberg. A census of protein repeats. *Journal of molecular biology*, 293(1) :151–60, October 1999.

[77] Juliette Martin, Guillaume Letellier, Antoine Marin, Jean-François Taly, Alexandre G de Brevern, and Jean-François Gibrat. Protein secondary structure assignment revisited : a detailed analysis of different assignment methods. *BMC structural biology*, 5 :17, January 2005.

[78] Rune Matthiesen. Methods, algorithms and tools in computational proteomics : a practical point of view. *Proteomics*, 7(16) :2815–32, August 2007.

[79] L Mavridis, V Venkatraman, D W Ritchie, N Morikawa, R Andonov, A Cornu, J Nicolas, M Reisert, H Burkhardt, A Axenopoulos, and P Daras. SHREC'10 Track : Protein Models. 2010.

[80] Shintaro Minami, Kengo Sawada, and George Chikenji. MICAN : a protein structure alignment algorithm that can handle Multiple-chains, Inverse alignments, C($\alpha$) only models, Alternative alignments, and Non-sequential alignments. *BMC bioinformatics*, 14(1) :24, January 2013.

[81] Kevin B Murray, William R Taylor, and Janet M Thornton. Toward the detection and validation of repeats in protein structure. *Proteins*, 57(2) :365–80, November 2004.

[82] Alexey G. Murzin. New protein folds. *Current Opinion in Structural Biology*, 4(3) :441–449, June 1994.

[83] Alexey G. Murzin, Steven E. Brenner, Tim Hubbard, and Cyrus Chothia. SCOP : A structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247(4) :536–540, April 1995.

[84] Sampo Niskanen and Patric R J Östergård. Cliquer User's Guide, Version 1.0. Technical report, Helsinki University of Technology, 2003.

[85] Malod-dognin No. MALOD-DOGNIN Noël. 2010.

[86] D L Ollis, E Cheah, M Cygler, B Dijkstra, F Frolow, S M Franken, M Harel, S J Remington, I Silman, and J Schrag. The alpha/beta hydrolase fold. *Protein engineering*, 5(3) :197–211, April 1992.

[87] A Orengo. From protein structure. pages 374–382.

[88] C. a. Orengo, a. M. Martin, G. Hutchinson, S. Jones, D. T. Jones, a. D. Michie, M. B. Swindells, and J. M. Thornton. Classifying a Protein in the CATH Database of Domain Structures. *Acta Crystallographica Section D Biological Crystallography*, 54(6) :1155–1167, November 1998.

[89] C. A. Orengo, A. M. Martin, G. Hutchinson, S. Jones, D. T. Jones, A. D. Michie, M. B. Swindells, and J. M. Thornton. Classifying a protein in the CATH database of domain structures. In *Acta Crystallographica Section D : Biological Crystallography*, volume 54, pages 1155–1167, 1998.

[90] C A Orengo, A D Michie, S Jones, D T Jones, M B Swindells, and J M Thornton. CATH–a hierarchic classification of protein domain structures. *Structure (London, England : 1993)*, 5 :1093–1108, 1997.

[91] C a Orengo, F M Pearl, J E Bray, a E Todd, a C Martin, L Lo Conte, and J M Thornton. The CATH Database provides insights into protein structure/function relationships. *Nucleic acids research*, 27(1) :275–9, January 1999.

[92] Christine A Orengo and Janet M Thornton. Protein families and their evolution-a structural perspective. *Annual review of biochemistry*, 74 :867–900, 2005.

[93] Angel R Ortiz, Charlie E M Strauss, and Osvaldo Olmea. MAMMOTH (Matching molecular models obtained from theory) : An automated method for model comparison. pages 2606–2621, 2002.

[94] Shashi Bhushan Pandit and Jeffrey Skolnick. Fr-TM-align : a new protein structural alignment method based on fragment alignments and the TM-score. *BMC bioinformatics*, 9 :531, January 2008.

[95] R Gonzalo Parra, Rocío Espada, Ignacio E Sánchez, Manfred J Sippl, and Diego U Ferreiro. Detecting repetitions and periodicities in proteins by tiling the structural space. *The journal of physical chemistry. B*, 117(42) :12887–97, October 2013.

[96] Frances Pearl, Annabel Todd, Ian Sillitoe, Mark Dibley, Oliver Redfern, Tony Lewis, Christopher Bennett, Russell Marsden, Alistair Grant, David Lee, Adrian Akpor, Michael Maibaum, Andrew Harrison, Timothy Dallman, Gabrielle Reeves, Ilhem Diboun, Sarah Addou, Stefano Lise, Caroline Johnston, Antonio Sillero, Janet Thornton, and Christine Orengo. The CATH domain structure database and related resources Gene3D and DHS provide comprehensive domain family information for genome analysis. *Nucleic Acids Research*, 33(DATABASE ISS.), 2005.

[97] Gregory A. Petsko and Dagmar Ringe. *Protein Structure and Function*. 2004.

[98] Eric F Pettersen, Thomas D Goddard, Conrad C Huang, Gregory S Couch, Daniel M Greenblatt, Elaine C Meng, and Thomas E Ferrin. UCSF Chimera–a visualization system for exploratory research and analysis. *Journal of computational chemistry*, 25(13) :1605–12, October 2004.

[99] Oliver C Redfern, Andrew Harrison, Tim Dallman, Frances M G Pearl, and Christine a Orengo. CATHEDRAL : a fast and effective algorithm to predict folds and domain boundaries from multidomain protein structures. *PLoS computational biology*, 3(11) :e232, November 2007.

[100] Jane Reece, Lisa A. Urry, Noel Meyers, Michael L. Cain, Steven A. Wasserman, Peter V. Minorsky, Robert B. Jackson, and Bernard N. Cooke. *Campbell Biology*. 2011.

[101] Saeed Salem, Mohammed J. Zaki, and Chris Bystroff. FlexSnap : Flexible non-sequential protein structure alignment. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5724 LNBI, pages 273–285, 2009.

[102] Saeed Salem, Mohammed J Zaki, and Chris Bystroff. FlexSnap : flexible non-sequential protein structure alignment. *Algorithms for molecular biology : AMB*, 5 :12, 2010.

[103] C Sander and R Schneider. Database of homology-derived protein structures and the structural meaning of sequence alignment. *Proteins*, 9(1) :56–68, January 1991.

[104] Tobias Schmidt-Goenner, Aysam Guerler, Bjoern Kolbeck, and Ernst Walter Knapp. Circular permuted proteins in the universe of protein folds. *Proteins : Structure, Function, and Bioinformatics*, 78(7) :1618–1630, May 2010.

[105] Stefan Schmitt, Daniel Kuhn, and Gerhard Klebe. A New Method to Detect Related Function Among Proteins Independent of Sequence and Fold Homology. *Journal of Molecular Biology*, 323(2) :387–406, October 2002.

[106] Maxim Shatsky, Ruth Nussinov, and Haim J Wolfson. FlexProt : alignment of flexible protein structures without a predefinition of hinge regions. *Journal of computational biology : a journal of computational molecular cell biology*, 11(1) :83–106, January 2004.

[107] I N Shindyalov and P E Bourne. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein engineering*, 11(9) :739–47, September 1998.

[108] Michael L Sierk and Gerard J Kleywegt. Déjà vu all over again : finding and analyzing protein structure similarities. *Structure (London, England : 1993)*, 12(12) :2103–11, December 2004.

[109] Manfred J Sippl and Markus Wiederstein. Detection of spatial correlations in protein structures and molecular complexes. *Structure (London, England : 1993)*, 20(4) :718–28, 2012.

[110] Alex W Slater, Javier I Castellanos, Manfred J Sippl, and Francisco Melo. Towards the development of standardized methods for comparison, ranking and evaluation of structure alignments. *Bioinformatics (Oxford, England)*, 29(1) :47–53, January 2013.

[111] D J States and W Gish. Combined use of sequence similarity and codon bias for coding region identification. *Journal of computational biology : a journal of computational molecular cell biology*, 1(1) :39–50, January 1994.

[112] Lin Wang, Ling-Yun Wu, Yong Wang, Xiang-Sun Zhang, and Luonan Chen. SANA : an algorithm for sequential and non-sequential protein structure alignment. *Amino acids*, 39(2) :417–25, July 2010.

[113] Sheng Wang, Jianzhu Ma, Jian Peng, and Jinbo Xu. Protein structure alignment beyond spatial proximity. *Scientific reports*, 3 :1448, January 2013.

[114] January Weiner and Erich Bornberg-Bauer. Evolution of circular permutations in multidomain proteins. *Molecular biology and evolution*, 23(4) :734–43, April 2006.

[115] Inken Wohlers, Mathilde Le Boudic-Jamin, Hristo Djidjev, GunnarW. Klau, and Rumen Andonov. Exact Protein Structure Classification Using the Maximum Contact Map Overlap Metric. In Adrian-Horia Dediu, Carlos Martín-Vide, and Bianca Truthe, editors, *Algorithms for Computational Biology*, volume 8542 of *Lecture Notes in Computer Science*, pages 262–273. Springer International Publishing, 2014.

[116] Inken Wohlers, Noël Malod-Dognin, Rumen Andonov, and Gunnar W Klau. CSA : comprehensive comparison of pairwise protein structure alignments. *Nucleic acids research*, 40(Web Server issue) :W303–9, July 2012.

[117] Willy Wriggers and Klaus Schulten. RESEARCH ARTICLES Protein Domain Movements : Detection of Rigid Domains and Visualization of Hinges in Comparisons of Atomic Coordinates. 14(September 1996) :1–14, 1997.

[118] Jinrui Xu and Yang Zhang. How significant is a protein structure similarity with TM-score = 0.5 ? *Bioinformatics (Oxford, England)*, 26(7) :889–95, April 2010.

[119] Yuzhen Ye and Adam Godzik. Database searching by flexible protein structure alignment. *Protein science : a publication of the Protein Society*, 13(7) :1841–1850, 2004.

[120] Yuzhen Ye and Adam Godzik. FATCAT : a web server for flexible structure comparison and structure similarity searching. *Nucleic acids research*, 32(Web Server issue) :W582–5, July 2004.

[121] J. Yon-Kahn. *Histoire de la science des protéines*. EDP sciences, 2006.

[122] Zheng Yuan, Timothy L Bailey, and Rohan D Teasdale. Prediction of protein B-factor profiles. *Proteins*, 58(4) :905–12, March 2005.

[123] a. Zemla. LGA : a method for finding 3D similarities in protein structures. *Nucleic Acids Research*, 31(13) :3370–3374, July 2003.

[124] Adam Zemla, Brian Geisbrecht, Jason Smith, Marisa Lam, Bonnie Kirkpatrick, Mark Wagner, Tom Slezak, and Carol Ecale Zhou. STRALCP–structure alignment-based clustering of proteins. *Nucleic acids research*, 35(22) :e150, January 2007.

[125] Yang Zhang and Jeffrey Skolnick. Scoring function for automated assessment of protein structure template quality. *Proteins*, 57(4) :702–10, December 2004.

[126] Yang Zhang and Jeffrey Skolnick. TM-align : a protein structure alignment algorithm based on the TM-score. 33(7) :2302–2309, 2005.

[127] Robert Zwanzig, Attila Szabo, and Biman Bagchi. Levinthal's paradox. *Proceedings of the National Academy of Sciences*, 89(1) :20–22, 1992.

# Annexes

# Annexe A

# Définitions générales

## Notions de chimie organique, définitions générales

Les notions développées dans cette section et la suivante sont tirées du Biology de Campbell, Reece *et al.* [100]. Les définitions générales sont restreintes aux cas rencontrés au sein des protéines.

### Propriétés atomiques

**Définition A.1 (Atome lourd)** *Dans le cas des protéines, atome lourd désigne tout atome (majoritairement **C**, **N**, **O**, **S**) qui ne soit pas un hydrogène (**H**).*

**Définition A.2 (Electronegativité)** *Capacité d'un atome à attirer les électrons. Plus un atome est électronégatif, plus il agit comme un attracteur d'électrons.*

**Définition A.3 (Polarité)** *Capacité à créer des liaisons électrostatiques avec des molécules d'eau ($H_2O$).*

**Définition A.4 (Hydrophobicité)** *Capacité d'un atome à repousser l'eau. Un composé hydrophobe ne contient pas de groupe chargé ou d'atome capable de former des liaisons hydrogène. Les molécules hydrophobes peuvent former des liaisons faibles.*

### Liaisons inter-atomiques

**Définition A.5 (Liaison covalente entre deux atomes)** *Mise en commun d'un ou plusieurs électrons. Lorsque la liaison a lieu entre atomes du même élément, la liaison est apolaire, polaire dans les autres cas.*

**Définition A.6 (Liaison ionique)** *Liaison entre deux atomes avec une trop forte différence d'électronégativité.*

**Définition A.7 (Liaison Hydrogène)** *Attraction entre un atome d'hydrogène et un atome électronégatif. Lorsqu'un atome d'hydrogène déjà lié à un atome subit l'attraction d'un second atome électronégatif (souvent **N** ou **O**), les deux atomes lourds se partagent l'hydrogène.*

### Définition de quelques molécules

**Définition A.8 (Hydrocarbure)** *Composé organique constitué uniquement d'atomes de carbone (**C**) et d'atomes d'hydrogène (**H**).*

**Définition A.9 (Amphiphatique ou amphiphile)** *Se dit d'une molécule qui contient à la fois une partie hydrophile et une partie hydrophobe.*

### Groupes fonctionnels chimiques

**Définition A.10 (Groupe fonctionnel chimique)** *Atome ou groupe d'atomes dont le comportement chimique est similaire (donc prévisible) pour les différents composés où il apparaît.*

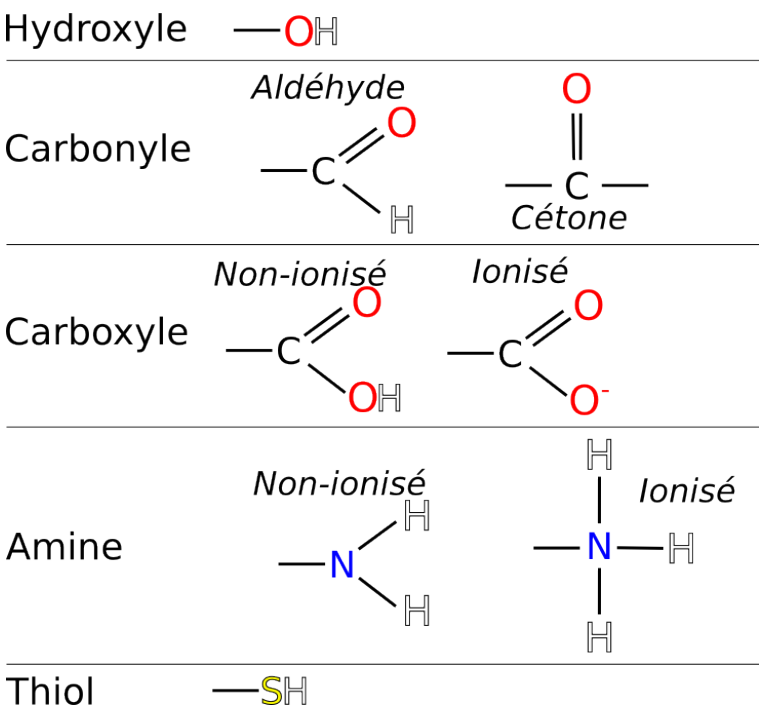



FIGURE A.1 – Formules des principaux groupes fonctionnels

**Définition A.11 (Groupe hydroxyle)** *Atome d'hydrogène (**H**) lié à un atome d'oxygène (**O**).*

**Définition A.12 (Groupe carbonyle)** *Est caractérisé par une double liaison entre un atome d'oxygène (**O**) et un atome de carbone (**C**), ce dernier étant autrement lié à des atomes de carbone (**C**) ou d'hydrogène (**H**) exclusivement.*

**Définition A.13 (Groupe carboxyle)** *Atome d'oxygène (**O**) uni par une double liaison à un carbone (**C**) lié à un groupement **hydroxyle**. Le groupement carboxyle a également des propriétés d'acide car il a tendance à s'ioniser en perdant un proton ($H^+$). Ce groupement est un constituant de la partie invariante des **acides aminés**(cf Section 1.3).*

**Définition A.14 (Groupe amine)** *Comprend un atome d'azote (**N**) lié ou non à un ou plusieurs atomes d'hydrogène (**H**), et à une chaîne carbonée.*

Les **acides aminés** comprennent un groupement **carboxyle** et un groupement **amine** (d'où leur nom).

**Définition A.15 (Groupe thiol)** *Atome de soufre (**S**) lié à une hydrogène et attaché à un radical.*

**Définition A.16 (Cycles aromatiques)** *Structure plane et stable. Les atomes peuvent s'associer et former des cycles, partageant ainsi des électrons qui, délocalisés, stabilisent l'ensemble.*
*L'**imidazole** est un cycle aromatique hétérocyclique (composé d'atomes de natures différentes) à cinq atomes.*

**Définition A.17 (Groupement aliphatique)** *Du grec alipheir (graisse) : hydrocarbure non-aromatique. Pour les protéines, le groupement aliphatique est restreint aux portions de chaîne latérale hydrocarbures saturées (avec donc des propriétés hydrophobes).*

## Notions de biologie

**Définition A.18 (Acide aminé)** *Acide carboxylique possédant entre autre un groupement **amine**.*

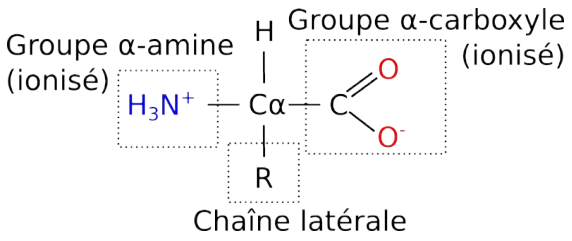



FIGURE A.2 – Formule générale des acides aminés.

**Définition A.19 (Résidu d'acide aminé)** *Partie non modifiée après insertion au sein du polypeptide.*

**Définition A.20 (Enzyme)** *Protéine qui catalyse (favorise) une ou plusieurs réactions chimiques (nom masculin).*

# Annexe B

# Notions de théorie des Graphes

Cette thèse est en partie centrée sur la modélisation de protéines sous forme de graphes. Une protéine peut être considérée comme un ensemble de points dispersés dans l'espace dont certains points sont étiquetés par diverses propriétés physico-chimiques. De même ces points sont répartis dans un espace euclidien à trois dimensions. Dans cette annexe nous allons décrire les notions de la théorie des graphes que nous utilisons pour caractériser les protéines et la comparaison de protéine.

## B.1 Graphe non-orienté

**Définition B.1 (Graphe non-orienté)** *Un graphe non-orienté $G = (V, E)$ (figure B.1) est composé d'un ensemble de sommets V et d'un ensemble d'arêtes E. Une arête $e_{(v,w)} \in E$ entre deux sommets $v, w \in V$ est non-orientée, c'est-à-dire : $e_{(v,w)} \leftrightarrow e_(w, v)$. La présence (respectivement absence) d'une arête $e_{(v,w)}$ se modélise par une variable binaire $x_{(v,w)} \in \{1, 0\}$ qui signifie : 1 présence , 0 sinon.*

Un graphe est une représentation abstraite d'un objet (ici de protéines) ou d'un problème. Le parcours de graphes permet d'extraire des informations ou des solutions. Nous recherchons des sous-ensembles de sommets du graphe avec des propriétés spécifiques : des cliques et des pseudo-cliques.

**Définition B.2 (Adjacence)** *Deux sommets d'un graphe sont adjacents s'ils sont connectés par une arête.*

**Définition B.3 ($k$-clique (ou clique de taille $k$))** *Ensemble de k sommets d'un graphe non-orienté tel que chaque couple de sommets est adjacent. La taille d'une clique (aussi appelé cardinal) est le nombre de sommets qu'elle contient. Une clique de taille $k$ implique un nombre d'arêtes égal à $\frac{k(k-1)}{2}$.*

**Définition B.4 (Clique maximum)** *Une clique maximum du graphe est une clique telle qu'il n'existe aucune autre clique dans le graphe avec un nombre de sommets supérieur.*

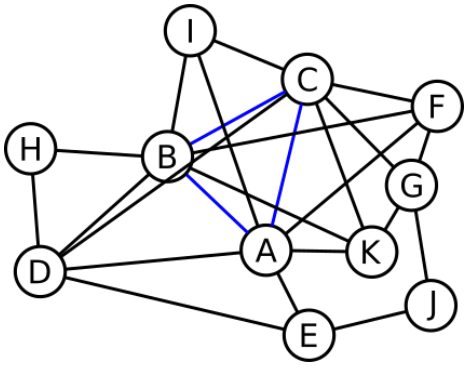



FIGURE B.1 – Exemple de graphe non-orienté. Les sommets, ou nœuds, sont nommés par des lettres et seules les arêtes existantes sont représentées. En bleu les arêtes d'une clique de taille 3

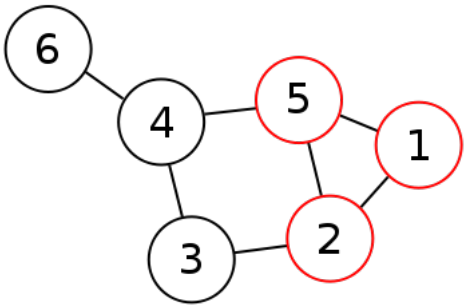



FIGURE B.2 – graphe non-orienté avec clique maximum (en rouge) de taille 3

Dans la figure B.2, la clique maximum (en rouge) est de taille 3. Nous remarquons d'autres cliques dans ce graphe, par exemple les sommets 3-4 forment une clique de taille 2, mais aucune n'est de taille supérieure à 3. Plusieurs cliques maximum peuvent exister au sein d'un graphe et posséder des sommets communs ou non. La recherche de cliques maximum au sein d'un graphe est un problème NP-difficile (Karp, 1972). Des algorithmes exacts existent ( e.g. [84]) mais la recherche exhaustive s'avère chronophage.

**Définition B.5 (Clique maximale)** *Une clique maximale est une clique qui ne peut être étendue par l'ajout d'un sommet.*

Une clique maximale peut ou non être une clique maximum mais sa propriété principale est qu'elle ne peut être étendue. Chaque sommet restant du graphe n'est pas connecté (i.e. il n'existe pas d'arête) avec tous les sommets de la clique maximale.

## B.2 Pseudocliques

Par souci de clarté, nous ne présentons ici que le type de pseudo-clique relatif à nos travaux.

**Définition B.6 (Pseudoclique)** *Ensemble de $k$ sommets connectés à moins de $\frac{k(k-1)}{2}$ arêtes.*

Dans cette thèse une pseudoclique est un ensemble de sommets tel que chaque sommet est connecté à tous les sommets d'une $k$-clique (nommée $k$-graine). Sauf cas exceptionnel, $k = 3$ (c.a.d. un triangle).

L'une des propriétés de la pseudo-clique est que tous ses sommets ne sont pas connectés entre eux. En effet, le critère d'ajout d'un sommet à la pseudo-clique est l'existence d'une arête entre le sommet considéré et les sommets de la $k$-clique. En revanche, deux sommets de la pseudo-clique non issus de la $k$-clique peuvent ne pas être connectés.

# Annexe C

# Comparaison d'outils sur le jeu de données MALIDUP-NS

MALIDUP-NS [30][80] est un jeu de données composé de 241 paires de structures dont l'une est modifiée pour contenir une permutation circulaire. Nous avons comparé nos outils ainsi que des outils de la littérature au niveau géométrique, l'une des méthodes les plus usitées pour dire qu'un outil est meilleur qu'un autre est de comparer les résultats en termes de longueur d'alignement et de RMSDc associé. Nous présentons les résultats pour 5 outils : CECP, GANGSTA, MICAN, SANA et le couple ShinobiNinjas(noté SN).

Table C.1 – Comparaisons de CECP, GANGSTA,MICAN,SANA et ShinobiNinjas(SN) sur MALDUP-NS, on compte le nombre de fois où chaque outil retourne un alignement plus long et un RMSDc inférieur ou égal que l'autre outil, le nombre de cas où les deux outils retournent les mêmes valeurs et les cas non-concluants

| Outil 1/Outil 2 | Outil 1 | Outil 2 | Egalité | Non-concluant |
|---|---|---|---|---|
| CECP/GANGSTA | 16 | 95 | 5 | 123 |
| CECP/MICAN | 5 | 129 | 4 | 99 |
| CECP/SANA | 1 | 173 | 4 | 58 |
| CECP/SN | 4 | 76 | 2 | 155 |
| GANGSTA/MICAN | 6 | 57 | 24 | 150 |
| GANGSTA/SANA | 5 | 158 | 10 | 65 |
| GANGSTA/SN | 1 | 43 | 3 | 227 |
| MICAN/SANA | 3 | 155 | 13 | 67 |
| MICAN/SN | 1 | 3 | 3 | 229 |
| SANA/SN | 2 | 4 | 3 | 227 |

Le tableau C.1 montre que la majorité des comparaisons ne permet pas de conclure, c'est à dire les résultats ne sont pas comparables. Une comparaison est non-concluante si un alignement est plus long qu'un autre mais son RMSDc est plus élevé. Un alignement est

dit meilleur que l'autre s'il est plus long et que son RMSDc est inférieur. Ici il est délicat de conclure sur la pertinence des outils, lequel est "meilleur" que les autres ? C'est incertain. CECP paraît moins performant que les autres, les outils comparés retournant plus souvent de meilleurs résultats. De même pour GANGSTA, mais entre SANA, SN et MICAN les résultats ne permettent pas de conclure. SANA est souvent meilleur que MICAN mais par rapport à SN le nombre de comparaisons non concluantes est majoritaire.

# Annexe D

# Résultats des analyses sur le jeu de données MALIDUP-NS

Ces résultats correspondent aux boxplots du chapitre 7. Ils montrent deux tendances : MICAN/ShiNi d'un côté et FlexSnap/TMalign de l'autre. Les outils rigides et tolérant les permutations ont des résultats se rapprochant fortement des valeurs de références.

TABLE D.1 – Dispersion des valeurs selon l'alignement

| ALI | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 22 | 12 | 8 | 12 | 22 |
| Q1 | 51 | 54 | 29 | 35 | 51 |
| MED | 69 | 76 | 48 | 49 | 75 |
| Q3 | 94 | 100 | 69 | 65 | 100 |
| MAX | 259 | 259 | 250 | 175 | 250 |

Table D.2 – Dispersion des valeurs selon le RMSDc

| RMSDc | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.576255 | 0.278662 | 0.66662 | 0.458466 | 0.66269 |
| Q1 | 1.84971 | 2.1443 | 1.72861 | 1.64864 | 1.84854 |
| MED | 2.28816 | 2.63931 | 1.99368 | 2.13346 | 2.23855 |
| Q3 | 3.04191 | 3.07143 | 2.37945 | 2.53317 | 2.5423 |
| MAX | 6.85239 | 4.16635 | 3.66281 | 3.41188 | 2.92396 |

Table D.3 – Dispersion des valeurs selon le RMSDd

| RMSDd | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.411853 | 0.265217 | 0.354096 | 0.320308 | 0.481583 |
| Q1 | 1.35523 | 1.54498 | 1.27871 | 1.22346 | 1.43142 |
| MED | 1.68272 | 1.96814 | 1.55928 | 1.67597 | 1.78512 |
| Q3 | 2.20339 | 2.27671 | 1.85372 | 2.00013 | 2.0527 |
| MAX | 6.24639 | 3.48173 | 2.75023 | 3.00406 | 2.40988 |

Table D.4 – Dispersion des valeurs selon le score MI

| MI | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.054798 | 0.0876113 | 0.0346427 | 0.0470272 | 0.0859662 |
| Q1 | 0.152206 | 0.164073 | 0.12889 | 0.129492 | 0.198897 |
| MED | 0.216354 | 0.208296 | 0.176914 | 0.173325 | 0.239514 |
| Q3 | 0.302857 | 0.276554 | 0.224434 | 0.249149 | 0.305056 |
| MAX | 1.01033 | 1.01036 | 0.671569 | 0.674393 | 0.982048 |

Table D.5 – Dispersion des valeurs selon le score SI

| SI | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.659851 | 0.659833 | 0.992998 | 0.989574 | 0.679258 |
| Q1 | 2.20269 | 2.4115 | 3.00011 | 2.69574 | 2.18788 |
| MED | 3.10847 | 3.20594 | 3.78659 | 3.85516 | 2.78906 |
| Q3 | 4.41742 | 4.0734 | 5.26036 | 5.23146 | 3.3682 |
| MAX | 12.2384 | 7.63344 | 20.2726 | 14.3805 | 7.85528 |

TABLE D.6 – Dispersion des valeurs selon le Qscore

| Q | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.0310848 | 0.0679224 | 0.00503429 | 0.0225962 | 0.0439645 |
| Q1 | 0.182842 | 0.213739 | 0.0804189 | 0.0995714 | 0.259334 |
| MED | 0.301176 | 0.327066 | 0.170916 | 0.160228 | 0.350007 |
| Q3 | 0.478952 | 0.479477 | 0.277348 | 0.260629 | 0.484944 |
| MAX | 0.942848 | 0.942848 | 0.829375 | 0.692124 | 0.925773 |

TABLE D.7 – Dispersion des valeurs selon le score SAS

| SAS | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.325379 | 0.32538 | 0.636537 | 0.474121 | 0.352915 |
| Q1 | 2.48811 | 2.4744 | 2.99969 | 3.11291 | 2.0544 |
| MED | 3.4413 | 3.43322 | 4.07009 | 4.21839 | 2.99673 |
| Q3 | 4.58906 | 4.48724 | 6.38752 | 5.69167 | 3.79447 |
| MAX | 19.5977 | 12.3008 | 24.7437 | 12.915 | 10.6123 |

TABLE D.8 – Dispersion des valeurs selon le RMSD100

| RMSD100 | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.523878 | -4.63419 | -31.2416 | -14.7671 | 0.566596 |
| Q1 | 2.32659 | 2.52457 | 2.52657 | 2.57069 | 2.11303 |
| MED | 3.01094 | 3.04799 | 3.2452 | 3.2949 | 2.63129 |
| Q3 | 3.91949 | 3.77317 | 4.81455 | 4.2962 | 3.1028 |
| MAX | 16.4203 | 15.2582 | 27.4649 | 16.8798 | 9.6104 |

TABLE D.9 – Dispersion des valeurs selon la SeqID

| Seqid | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 1.72414 | 2.43902 | 0 | 0 | 0 |
| Q1 | 10.1449 | 10.1852 | 7.27273 | 8.21918 | 7.44681 |
| MED | 15.1899 | 14.4144 | 12.3288 | 12.8713 | 11.7647 |
| Q3 | 21.4286 | 20.2381 | 18.617 | 19.2308 | 18.617 |
| MAX | 72.2467 | 72.2467 | 60 | 66.4706 | 72.1239 |

Table D.10 – Dispersion des valeurs selon le TMscore (P1)

| TMP1 | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.215817 | 0.187575 | 0.0711058 | 0.160215 | 0.189067 |
| Q1 | 0.420531 | 0.451758 | 0.264327 | 0.304583 | 0.475495 |
| MED | 0.531154 | 0.565567 | 0.393857 | 0.374548 | 0.563084 |
| Q3 | 0.652098 | 0.672006 | 0.524823 | 0.474194 | 0.664404 |
| MAX | 0.98456 | 0.98456 | 0.924186 | 0.805219 | 0.978088 |

Table D.11 – Dispersion des valeurs selon le TMscore (P2)

| TMP2 | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.174686 | 0.260518 | 0.074299 | 0.166103 | 0.211567 |
| Q1 | 0.404483 | 0.453126 | 0.276032 | 0.297693 | 0.458309 |
| MED | 0.528895 | 0.561771 | 0.394315 | 0.378188 | 0.564364 |
| Q3 | 0.654036 | 0.685229 | 0.511273 | 0.488506 | 0.673927 |
| MAX | 0.98456 | 0.98456 | 0.924186 | 0.85267 | 0.978088 |

Table D.12 – Dispersion des valeurs selon le TMscore (moyen)

| Tmmoy | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.205441 | 0.226683 | 0.0745694 | 0.167171 | 0.217337 |
| Q1 | 0.418548 | 0.450948 | 0.271443 | 0.300017 | 0.467608 |
| MED | 0.525424 | 0.561576 | 0.383319 | 0.373527 | 0.562067 |
| Q3 | 0.650042 | 0.676327 | 0.51019 | 0.478408 | 0.67022 |
| MAX | 0.98456 | 0.98456 | 0.924186 | 0.831672 | 0.978088 |

Table D.13 – Dispersion des valeurs selon le Zscore

| Zscore | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | -1.7 | -1.61792 | -1.47033 | -1.432 | -0.987662 |
| Q1 | -0.705295 | -0.758193 | 0.686734 | 0.442023 | 1.42096 |
| MED | 2.37344 | 0.733184 | 2.848 | 2.56741 | 3.89955 |
| Q3 | 7.05464 | 3.96052 | 5.22517 | 5.88015 | 7.3938 |
| MAX | 90.0475 | 90.0474 | 50.7562 | 64.2759 | 86.4721 |

TABLE D.14 – Dispersion des valeurs selon le Sscore

| Sscore | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 12.6239 | 17.0034 | 8.05504 | 16.5033 | 19.7918 |
| Q1 | 47.407 | 48.2499 | 30.8615 | 36.6896 | 52.3172 |
| MED | 59.6307 | 61.8779 | 47.5254 | 47.7718 | 69.0298 |
| Q3 | 83.8695 | 83.998 | 66.6791 | 61.3463 | 97.7466 |
| MAX | 391.692 | 391.692 | 229.414 | 282.391 | 377.172 |

TABLE D.15 – Dispersion des valeurs selon le GSASscore

| GSAS | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 0.329737 | 0.329737 | 0.640838 | 0.476926 | 0.360899 |
| Q1 | 2.63134 | 2.59409 | 3.17438 | 3.40591 | 3.06359 |
| MED | 3.61276 | 3.65069 | 4.19648 | 4.76921 | 4.51598 |
| Q3 | 4.96229 | 4.75715 | 6.84378 | 6.76311 | 10.7107 |
| MAX | 20.4498 | 12.9482 | 24.7437 | 19.618 | 190.959 |

TABLE D.16 – Dispersion des valeurs selon la Normsim

| Normsim | Reference | MICAN | FlexSnap | TMalign | ShinobiNinjas |
|---|---|---|---|---|---|
| MIN | 32 | 27 | 8 | 21 | 26 |
| Q1 | 59 | 64 | 34 | 39 | 65 |
| MED | 71 | 78 | 50 | 48 | 73 |
| Q3 | 84 | 87 | 66 | 60 | 82 |
| MAX | 100 | 100 | 95 | 87 | 99 |

## Résumé

Cette thèse s'articule autour de la détection de similarités globales et locales dans les structures protéiques. Premièrement les structures sont comparées, mesurées en termes de distance métrique dans un but de classification supervisée. Cette classification des domaines structuraux au sein de classifications hiérarchiques se fait par le biais de dominances et d'apprentissages permettant d'assigner plus rapidement et de manière exacte de nouveaux domaines. Deuxièmement, nous proposons une méthode de manière à traduire un problème biologique dans le formalisme des graphes. Puis nous résolvons ce problème avec des algorithmes de parcours de ce graphes pour extraire les différentes sous-structures similaires. Cette méthode repose sur des notions de compatibilités entre éléments des structures ainsi que des critères de distances entre éléments. Ces techniques sont capables de détecter des événements tels que des permutations circulaires, des charnières (flexibilité) et des répétitions de motifs structuraux. Finalement nous proposons une nouvelle approche dans l'analyse fine de structures afin de faciliter la recherche de régions divergentes entre structures 3D fortement similaires.



## Mots-clefs

Classification supervisée de protéines, apprentissage, dominances, formalisation par des graphes, recherche de cliques, recherche de pseudocliques, analyse d'alignements structuraux, domaines structuraux, permutations circulaires, charnières, répétitions de motifs.



## Summary

This thesis focusses on local and global similarities and divergences inside protein structures. First, structures are scored, with criteria of similarity and distance in order to provide a supervised classification. This structural domain classification inside existing hierarchical databases is possible by using dominances and learning. These methods allow to assign new domains with accuracy and exactly. Second we focusses on local similarities and proposed a method of protein comparison modelisation inside graphs. Graph traversal allows to find protein similar substructures. This method is based on compatibility between elements and criterion of distances. We can use it and detect events such that circular permutations, hinges and structural motif repeats. Finally we propose a new approach of accurate protein structure analysis that focused on divergences between similar structures.



## Keywords

Supervised classification, learning, dominances, graph formalism, clique/pseudoclique detection, structural alignments, structural domains, circular permutations, hinges, structural motif repeats.