De novo detection of structure repeats in Proteins

Mathilde Le Boudic-Jamin, Rumen Andonov

De novo detection of structure repeats in Proteins

Mathilde LE BOUDIC-JAMIN[1] and Rumen ANDONOV[1]

IRISA UMR 6074 et Université de Rennes 1, Campus de Beaulieu, 263 avenue du Général Leclerc 35 042 RENNES cedex, France

Corresponding Author: mathilde.le_leboudic-jamin@inria.fr

Abstract *Almost 25% of proteins contains internal repeats, these repeats may have a major role in the protein function. Furthermore some proteins actually are the same substructure repeated many times, these proteins are solenoids. But only few repeat detection programs exist, we present here Kunoichi, a simple and efficient tool for discovering protein repeats. Kunoichi is based on protein fragment comparison and clique detection. As first results, we show that Kunoichi can find different levels of repetitions and successfully identify protein tiles. Kunoichi is available on request from the authors.*

Keywords Internal repeats, Solenoids, Protein structure, Fragment graph

Détection *de novo* de structures répétées au sein des protéines

Résumé *Environ un quart des protéines contiennent des répétitions internes, ces répétitions peuvent jouer un rôle crucial dans la fonction de leurs protéines.De plus, certaines protéines ne sont en fait qu'une succession d'une sous-structure répétées plusieurs fois. Ces protéines modulaires sont appelées protéines solénoïdes. Néanmoins, peu d'outils de détection de répétitions existent. Nous présentons ici Kunoichi, un outil de détection de répétitions simple et efficace. Kunoichi est basé sur la comparaison des fragments ainsi que sur la détection de cliques. Les premiers résultats montrent que Kunoichi peut trouver différents niveaux de répétitions et identifier avec succès les "tuiles" composants les protéines multi-répétées. Kunoichi est disponible sur demande auprès des auteurs.*

Mots-clés Répétitions internes, Solénoïdes, Structure protéique,Graphe de fragments

1. Introduction

La proportion de répétitions au sein des protéines est estimée à 25 % [11]. Ces répétitions, allant de quelques segments d'acides aminés à de larges domaines sont issues de duplications et de recombinaisons de portions de gènes [3]. 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 [10]. 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 [17]. D'où l'apparition d'outils de détection de séquences répétées tels que REPRO [6] ou IRIS [9] ou encore T-REKS [8] pour 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 ($\tilde{1}5\%$) ce qui a justifié le développement d'outils de détection de répétitions basé sur la structure tertiaire tels DAVROS[13], SWELFE[1], ConSole[7], PRIGSA[4] ou encore l'outil de dallage (« *tessellation* ») de

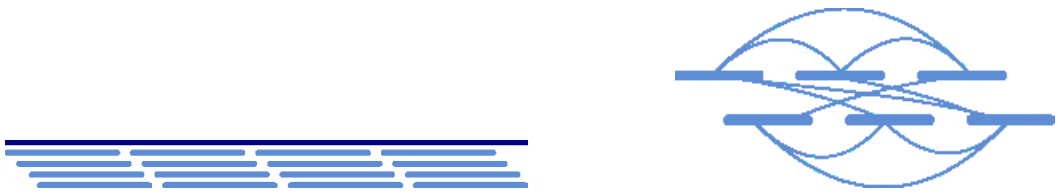
Parra *et al.*[15]. Nous avons développé *𝒦unoichi* , un outil de détection de répétitions exactes basé sur les structures primaires, secondaires et tertiaires des protéines. *𝒦unoichi* repose sur la modélisation des protéines en graphes de fragments et sur la recherche du plus grand ensemble de fragments compatibles pour un nombre de répétitions donné. Nous définissons ici le nombre de répétitions comme étant le nombre de motifs structuraux répétés au sein d'une protéine.

2. Méthodes

Graphe de fragments Notre méthodologie repose principalement sur la modélisation d'une protéine (P) sous forme d'un graphe de fragments (figure FIG. 1) et sur la notion de compatibilité entre fragments. Le **graphe de fragments** $G(P) = (F, E)$ associé à une protéine (P) est un *graphe non-orienté avec F l'ensemble des fragments continus chevauchants de longueur k et E l'ensemble des arêtes reliant les fragments compatibles*. Un **fragment** est *un ensemble de k résidus contigus de la protéine*, la protéine est scindée en $N-k$ fragments continus chevauchants avec N la longueur de la protéine et k la longueur souhaitée des fragments ce qu'illustre la figure FIG. 1. Deux fragments sont dits **compatibles** s'ils sont *non-chevauchants* et si leur *RMSDc de superposition est inférieur à un seuil τ*. La valeur de τ est un paramètre utilisateur (par défaut, $\tau = 3.0$Å). L'algorithme suivant (1) résume le procédé de création du graphe de fragments par *𝒦unoichi* .

Algorithm 1 graph_creation(P, τ, k)
avec P, une protéine de longueur N, τ, valeur limite de RMSDc entre fragments et k la longueur des fragments à créer.

$Residues(P) = \{Residue_1, \ldots, Residue_N\}$ ▷ Ensemble des résidus de la protéine
$Fragments = \{\}$ ▷ Ensemble des fragments continus
for $i = 0; i < |Residues|; i++$ **do**
  $CurrentFragment = Residues[i : i + k]$ ▷ Création du fragment commençant au $i^{ème}$ résidu et de longueur k
  $Fragments = Fragments \cup \{CurrentFragment\}$ ▷ Ajout du fragment courant
end for
$Edges = \{\}$ ▷ Ensemble des arcs
for $f = 0; f < |Fragments|; f++$ **do**
  Fragment = Fragments[f] ;
  for $g = f + 1; g < |Fragments|; g++$ **do**
    AnotherFragment = Fragments[g] ;
    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 $(Fragments, Edges)$



Découpage d'une protéine en fragments continus de taille k Graphe de fragments

Figure 1. Graphe de fragments

Résolution du graphe de fragments La résolution du graphe s'effectue à l'aide d'un solveur de cliques (dans notre cas *Cliquer* d'Ostergaard [14] mais tout autre solveur est utilisable) pour lequel nous demandons la recherche de toutes les cliques maximum de taille R. Chaque clique correspond à un ensemble de fragments non chevauchants répétés R fois.

Analyse des ensembles obtenus Trois cas sont possibles :

— le solveur ne trouve pas de clique correspondant aux paramètres d'entrée, ce qui signifie qu'il n'existe pas R répétitions de fragments géométriquement proches (à τÅ près) au sein de la protéine.
— le solveur trouve une seule clique donc un seul ensemble de fragments répétés.
— le solveur trouve plusieurs cliques soit plusieurs ensembles de fragments répétés.

Dans le dernier cas nous souhaitons choisir un ensemble de fragments dit « meilleur » que les autres. Pour cela nous calculons plusieurs scores pour chaque ensemble dont le pourcentage d'identité de séquence moyen et le pourcentage d'identité de structure secondaire moyen ($\overline{SSE}(f, f'), f\&f' \in F$) défini comme suit :

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

avec $SSE(f_i, f'_i) = 1$ si les résidus f_i et f'_i appartiennent à la même structure secondaire, 0 sinon. Le **meilleur ensemble de fragments** est alors celui ayant le *meilleur pourcentage d'identité de structure secondaire moyen et, en cas d'égalité, le meilleur pourcentage d'identité de séquence moyen.*

a. Recherche orientée de répétitions

Ce protocole 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. Ces paramètres permettent de différencier la recherche de répétitions en tandem du dallage de protéine. 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 fragments candidates. La longueur des fragments k est contrainte par l'inégalité suivante :

$$\left\lceil \frac{C * N}{R} \right\rceil \leq k \leq \left\lfloor \frac{N}{R} \right\rfloor \quad (2)$$

avec $k_{max} = \left\lfloor \frac{N}{R} \right\rfloor$ la longueur maximale que le fragment correspondant à l'unité répétée peut atteindre et $k_{min} = \left\lceil \frac{C*N}{R} \right\rceil$ la longueur minimale des fragments pour couvrir $C\%$ de la protéine une fois assemblés. L'utilisateur définit également une longueur de fragment minimale $\underline{k}$ « seuil ». Par conséquent, $k_{min} = max\left(\underline{k}, \left\lceil \frac{C*N}{R} \right\rceil\right)$. *Kunoichi* lance le module de création de graphe suivi du solveur de cliques et du module d'analyses pour la taille de fragment la plus élevée et en cas d'absence de solution recommence en diminuant la longueur du fragment jusqu'à obtention d'une solution ou avoir parcouru l'ensemble des longueurs de fragments possibles comme le montre l'agorithme 2.

Algorithm 2 search(P,R,τ,C,$\underline{k}$)
avec P, une protéine de longueur N, R le nombre de répétitions recherchées, τ limite de superposition de fragments, C pourcentage de recouvrement minimal attendu de la protéine et $\underline{k}$ la longueur minimale absolue des fragments.

$k_{max} = \left\lfloor \frac{N}{R} \right\rfloor$ ▷ Taille maximale des fragments autorisés
$k_{min} = max(\underline{k}, \left\lceil \frac{C*N}{R} \right\rceil)$ ▷ Taille minimale des fragments autorisés
for $k = k_{max}; k \geq k_{min}; k--$ **do**
  graphe_courant=graph_creation(P, τ, k); ▷ Création du graphe de fragments avec *Kunoichi*
    ▷ Recherche de toutes les cliques (all) maximales (max) de taille au moins égale à R dans le graphe courant
  clique_set=solve(all, max, R, graphe_courant) ▷ stockage des cliques trouvées
  if clique_set not empty **then**
    return clique_set and k; ▷ retourne les cliques trouvées pour une taille de fragment k
  end if
end for

La principale faiblesse de cet algorithme (discutée plus longuement dans la section 4.) est la nécessité de connaître à l'avance le nombre de répétitions recherchées. Nous l'avons donc un peu modifié pour obtenir un algorithme de détection *de novo* de répétitions.

b. Détection *de novo* de répétitions dans les structures protéiques

Etant donnée une structure protéique, nous allons tester d'un nombre variable de répétitions. Cette approche se rapproche de la méthode de Parra *et al.*[16] dans le sens où nous allons retomber sur des « tuiles » répétées recouvrant la protéine. Cela dit, la proportion de protéine recouverte par les fragments est chez nous un paramètre ainsi que le nombre de répétitions recherchées ce qui oriente les algorithmes dans leurs fractionnement des protéines ainsi que dans la recherche des éléments répétés.

Pour chaque nombre de répétitions testé, nous allons créer les fragments de taille correspondante et vérifier leur similarité. Ce protocole est présenté par l'algorithme 3.

Algorithm 3 *Kunoichi* (P,τ,C,$\underline{k}$)
avec P une protéine de longueur N, τ le seuil de superposition des fragments (défaut : 3.0Å) , C le pourcentage de recouvrement minimal ($défaut$: 80.0%) et $\underline{k}$ la longueur minimale absolue des fragments (défaut : 8 résidus)

$R_{max} = \left\lfloor \frac{N}{\underline{k}} \right\rfloor$ ▷ 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}
  search(P,R,τ,C,$\underline{k}$)
end for

Ce nouveau protocole retourne l'ensemble des répétitions acceptables (c-à-d répondant aux critères donnés) d'une protéine. Le protocole ne cherche que les ensembles de fragments maximaux pour chaque nombre de répétitions, par conséquent, il est orienté « dallage » de protéine.

3. Résultats

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 FIG. 2) se décompose en agencements d'hélices α et de feuillets β répétés. Parra *et al.* [15] 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 [13]. D'autre part, en appliquant l'outil ConSole de Hrabe et Godzik [7] nous avons détecté 15 répétitions de longueur 28 au sein de cette protéine.

Table 1. Recherche orientée de répétitions chez **2BNH,A**

Nb répétitions	τ(Å)	Recouvrement (%)	Nb résultats trouvés	Taille de fragments
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

Résultats de la recherche orientée Nous nous sommes basé sur les résultats précédemment évoqués pour rechercher des répétitions avec *Kunoichi* , le tableau TABLE 1 résume les résultats de l'algorithme 2 (recherche orientée de répétitions) pour la recherche de 8 et 15 répétitions au sein de **2BNH,A**. La recherche de 15 répétitions a induit des valeurs de longueur de fragments de taille variant entre 30 et 25, ce avec $\tau = 3.0$Å. Les premières recherches (pour $k = 30, 29, 28$) furent infructueuses mais pour $k = 27$ le protocole a bien trouvé 15 fragments superposables à 3.0Å visualisés sur la figure FIG. 2.

Table 2. Pourcentages d'identtié de structure secondaire et de séquence des neuf cliques contenant 15 répétitions de fragments de longueur 27 chez **2BNH,A**

Clique	$\overline{SSE_{id}}(\%)$	$\overline{Seq_{id}}(\%)$
Clique 1	72.143	27.687
Clique 2	72.619	27.109
Clique 3	72.619	29.422
Clique 4	74.524	28.367
Clique 5	74.524	28.061
Clique 6	74.524	29.728
Clique 7	74.626	29.456
Clique 8	74.245	29.524
Clique 9	74.218	29.218

On remarque qu'avec un seuil de RMSDc de superposition de fragments assez faible (3.0 Å), *𝒦unoichi* ne retrouve pas les résultats de Parra *et al.*, cependant, en augmentant ce seuil d'un point les résultats concordent mieux même si la longueur de fragments continue à différer (57 contre 48 pour nous). Cela s'explique par une flexibilité plus importante de leur outil, flexibilité que *𝒦unoichi* peut exprimer si l'on augmente son seuil de superposition.

Et en effet si le seuil est placé à 5.0 Å la longueur de fragments 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, *𝒦unoichi* retrouve plusieurs résultats, la figure Fig. 3 présente la répartitions des résidus selon les cliques et le tableau Table 2 résume 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 72 et 74% tandis que les pourcentages moyens d'identité de séquences sont faibles (entre 27 et 29%). Les résultats sont très similaires, au niveau des valeurs, le numéro 7 a un $\overline{SSE_{id}} = 74.626\%$ (contre 74.245% 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.

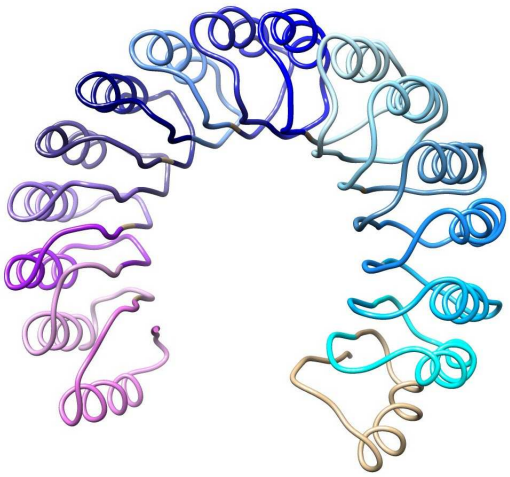


Figure 2. Représentation des 15 fragments au sein de **2BNH**.
En dégradé du bleu au rose sont représentés les fragments (un par couleur), en beige on retrouve les résidus qui n'appartiennent à aucune répétition.

Détection de novo de répétitions Nous avons testé le protocole décrit par l'algorithme 3 avec un recouvrement à 80.0 % et un RMSDc de superposition de fragments à 3.0Å sur **2BNH,A**. Le tableau Table 3 résume les résultats obtenus, *𝒦unoichi* détecte plusieurs niveaux de répétitions, la protéine est presque parfaitement recouverte par deux fragments identiques avec une déviation à 3.0Ået peut se découper en plus petits fragments 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 fragments est trop petite. D'autres répétitions peuvent exister mais à un niveau plus local, ici le protocole ne permet de découvrir que les fragments qui recréent la protéine lorsque mis bout à bout. Autre exemple avec deux protéines un peu différentes, d'un côté **1FQ0**, qui est un « *TIM barrel* » , une protéine contenant une alternance huit hélices α et huit brins β parallèles le long du squelette. De l'autre côté **1TL2,A** un « *β-propeller* » (protéine composée de feuillets β répétés et centrés).

L'étude *de novo* de ces structures a permis de détecter avec succès des sous-structures répétées significatives. Dans le cas de **1FQ0,A**, le fragment, de longueur 20, répété (huit fois) est composé d'un feuillet suivi d'une boucle et d'une hélice comme le montre la figure Fig. 5. La structure secondaire est moins conservée que ce qui était attendu (52.38%) et les séquences sont très divergentes (identité moyenne égale à 9.01%). Quant à **1TL2,A**, le fragment répété cinq fois (de longueur 46) est une sous-structure basée sur des feuillets β. Ces fragments sont très similaires au niveau de leur structure (comme attendu), pour un RMSDc

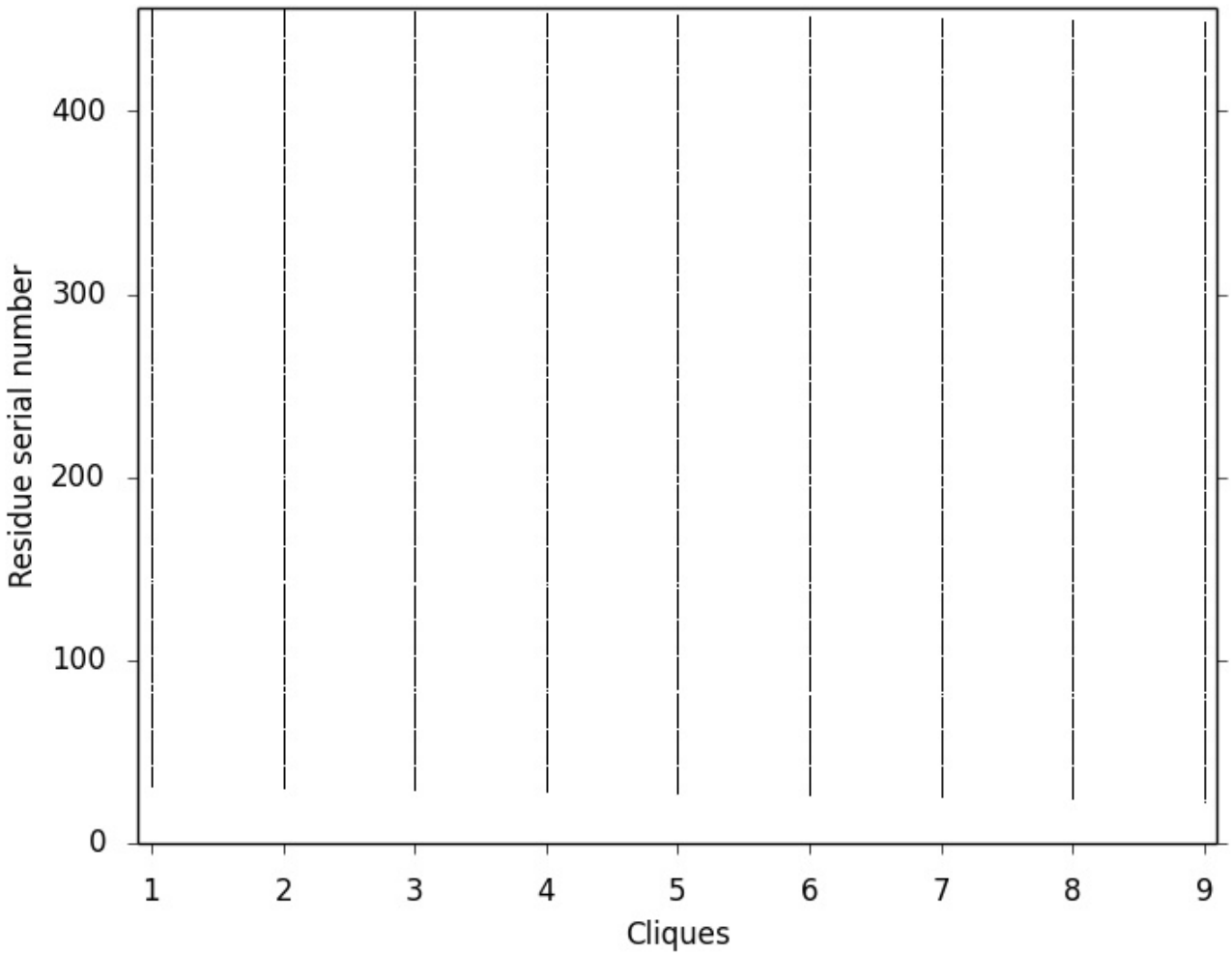
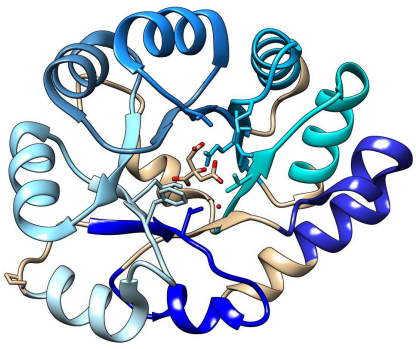
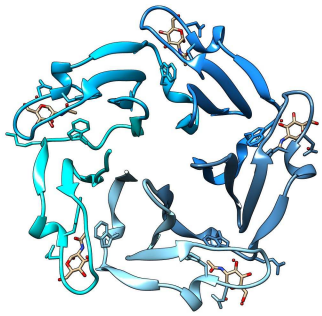


Figure 3. Analyse des neuf cliques trouvées chez **2BNH,A** Avec $\tau = 3.0$Å, une couverture = 80% et un nombre de répétitions égal à 15. Les pointillés noirs correspondent aux résidus appartenant à chaque clique.

limite à 4.0Å, 85.96 % en moyenne mais également au niveau de la similarité de séquence (53.40%). Cela montre assez bien la diversité des protéines répétées et la flexibilité qu'il peut y avoir entre deux fragments « identiques ».



1FQ0,A (8 fragments)



1TL2,A (5 fragments)

Figure 5. **1FQ0,A**(à g.) et **1TL2,A** (à d.)
En dégradé les différents fragments (un par couleur), en beige on retrouve les résidus qui n'appartiennent à aucune répétition.

Table 3. Détection de répétitions à 3.0 Å chez **2BNH,A**

Taille des fragments	# répétitions	Recouvrement (%)
225	2	98.684
141	3	92.763
102	4	89.474
84	5	92.105
56	7	85.965
27	14	82.895
27	15	88.816
24	16	84.211

4. Discussion

Nous avons donc créé une méthode entièrement automatique de détection de répétitions structurelles au sein des protéines. Néanmoins, cette méthode a quelques lacunes clairement identifiées :

— *Kunoichi* travaille entièrement avec des fragments 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 le solveur 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 *Samourai* par exemple) mais également au sein de chaque ensemble. Un alignement multiple (de séquence comme de structure) est envisagé pour mieux étudié ces répétitions.

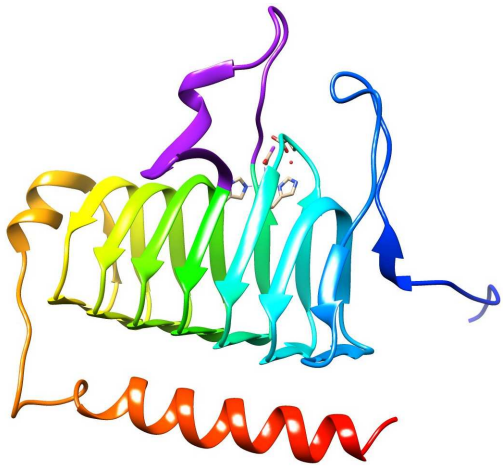


Figure 4. Représentation 3D de **1QRL**, en violet la boucle insérée au sein de la structure répétée

La continuité des fragments 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 β-solénoïdes comme le montre la figure Fig. 4. Les fragments 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 à cette lacune, l'une d'elle étant de combiner *Kunoichi* à *Shinobi* et *Ninjas* [5] lors de la création de fragments. Les fragments continus seraient alignés avec *Ninjas* puis transformés en fragments discontinus (déjà liés les uns autres). Et ces fragments discontinus reliés seront ensuite analysés par un solveur de cliques pour trouver le plus grand ensemble compatible comme dans la version classique de *Kunoichi* . Une autre piste consiste à modifier la fonction booléenne compatibility() du module de création de graphes en y incorporant un outil d'alignement de structures (ex : *Shinobi Ninjas* , TM-Align [18], A_purva [2] ou MICAN[12]). Un tel outil permettrait de trouver un alignement optimal qui, s'il est "bon", validera la paire de fragments. L'intérêt ici est d'ajouter une légère flexibilité avec des gaps.

5. Conclusion

Nous avons développé un outil simple et efficace pour détecter des répétitions au sein des protéines, *Kunoichi* est particulièrement adapté à la détection de solénoïdes mais permet aussi de détecter plusieurs niveaux de répétitions. La continuité des fragments est un point faible mais plusieurs pistes

en cours d'étude pour pallier ce défaut sont prometteuses. En conclusion l'étude des répétitions internes des protéines est un sujet encore ouvert, il reste de nombreux cas difficiles à étudier et de recherches à mener pour obtenir un outil complet efficace.

Références

[1] A.-L. Abraham, E. P. C. Rocha, and J. Pothier. Swelfe : a detector of internal repeats in sequences and structures. *Bioinformatics (Oxford, England)*, 24(13) :1536–7, July 2008.

[2] R. Andonov, N. Malod-Dognin, and N. Yanev. Maximum contact map overlap revisited. *J Comput Biol*, 18(1) :27–41, 2011.

[3] 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.

[4] B. Chakrabarty and N. Parekh. PRIGSA : protein repeat identification by graph spectral analysis. *Journal of bioinformatics and computational biology*, 12(6) :1442009, Dec. 2014.

[5] G. Chapuis, M. Le Boudic-Jamin, R. Andonov, H. Djidjev, and D. Lavenier. Parallel seed-based approach to multiple protein structure similarities detection. *Scientific Programming*, 2014.

[6] R. A. George and J. Heringa. The REPRO server : finding protein internal sequence repeats through the Web. *Trends in biochemical sciences*, 25(10) :515–517, 2000.

[7] T. Hrabe and A. Godzik. ConSole : using modularity of Contact maps to locate Solenoid domains in protein structures. *BMC bioinformatics*, 15(1) :119, 2014.

[8] J. Jorda and A. V. Kajava. T-REKS : identification of Tandem REpeats in sequences with a K-meanS based algorithm. *Bioinformatics (Oxford, England)*, 25(20) :2632–8, Oct. 2009.

[9] H.-Y. Kao, W.-S. Tzou, Y.-C. Hsu, C.-M. Chen, and T.-W. Pai. IRIS : Internal Repeat Identification System. 2009.

[10] B. Kobe and A. V. Kajava. When protein folding is simplified to protein coiling : the continuum of solenoid protein structures. *Trends in Biochemical Sciences*, 25(10) :509–515, Oct. 2000.

[11] E. M. Marcotte, M. Pellegrini, T. O. Yeates, and D. Eisenberg. A census of protein repeats. *Journal of molecular biology*, 293(1) :151–60, Oct. 1999.

[12] S. Minami, K. Sawada, and G. Chikenji. MICAN : a protein structure alignment algorithm that can handle Multiple-chains, Inverse alignments, C(α) only models, Alternative alignments, and Non-sequential alignments. *BMC bioinformatics*, 14(1) :24, Jan. 2013.

[13] K. B. Murray, W. R. Taylor, and J. M. Thornton. Toward the detection and validation of repeats in protein structure. *Proteins*, 57(2) :365–80, Nov. 2004.

[14] S. Niskanen and P. R. J. Östergård. Cliquer User's Guide, Version 1.0. Technical report, Helsinki University of Technology, 2003.

[15] R. G. Parra, R. Espada, I. E. Sánchez, M. J. Sippl, and D. U. Ferreiro. Detecting repetitions and periodicities in proteins by tiling the structural space. *The journal of physical chemistry. B*, 117(42) :12887–97, Oct. 2013.

[16] R. G. Parra, R. Espada, I. E. Sánchez, M. J. Sippl, and D. U. Ferreiro. Detecting repetitions and periodicities in proteins by tiling the structural space. *The journal of physical chemistry. B*, 117(42) :12887–97, Oct. 2013.

[17] C. Sander and R. Schneider. Database of homology-derived protein structures and the structural meaning of sequence alignment. *Proteins*, 9(1) :56–68, Jan. 1991.

[18] Y. Zhang and J. Skolnick. TM-align : a protein structure alignment algorithm based on the TM-score. 33(7) :2302–2309, 2005.