



École Doctorale INTERFACES
Approches interdisciplinaires: fondements, applications et
innovations

Titre de la thèse

Computational protein design: a tool for protein engineering and synthetic biology

présentée et soutenue publiquement le XXX

pour l'obtention du

Doctorat de l'Université Paris-Saclay

spécialité: Les sciences du vivant

par

M. David MIGNON

Composition du jury

Rapporteurs :	Dr. Prénom1 NOM1	Rapporteur externe
	Dr. Prénom2 NOM2	Rapporteur externe
	Pr. Prénom3 NOM3	Rapporteur interne

Examinateurs :	Dr. Prénom4 NOM4	Examinateur
	Dr. Prénom5 NOM5	Directeur de thèse
	Dr. Prénom6 NOM6	Directeur de thèse

Remerciements

XXX

à XXX.

Table des matières

Liste des figures	ix
Liste des tables	xiii
Abreviations	xv
Introduction	1
1 le « CPD » : Conception de protéine par ordinateur	5
1.1 l'espace des séquences-conformations	6
1.1.1 l'état replié	6
les chaînes latérales	7
Le squelette	7
1.1.2 l'état deplié	8
1.2 l'énergie de conformation	8
1.2.1 La mécanique moléculaire	8
1.2.2 Les interactions liées	9
1.2.3 Les interactions non liées	10
1.2.4 D'autres approches	11
1.3 Modélisation du solvant	11
1.3.1 modèle de solvant explicite	11
1.3.2 modèle implicite	13
1.3.3 modèle CASA	14
1.3.4 modèle Poisson Boltzmann	14
1.3.5 modèle Born Généralisé	15
1.4 l'algorithme d'exploration	16
1.4.1 Algorithme du champ moyen	18
1.4.2 Dead-End Elimination	18
1.4.3 le CFN	20

Table des matières

1.4.4	l'heuristique Multistart Steepest Descent (MSD)	23
1.4.5	Algorithme génétique	24
1.4.6	Monte-Carlo	25
1.5	Les programmes CPD	26
1.6	Les succès	26
2	proteus	27
3	Comparing stochastic search algorithms for computational protein design	29
3.1	Introduction	31
3.2	Methods	33
3.2.1	Monte Carlo : general framework	33
3.2.2	MC and REMC : implementation details	35
3.2.3	Heuristic sequence optimization	36
3.2.4	Cost function network method	36
3.2.5	Energy function	36
3.2.6	Test systems and preparation	37
3.2.7	Sequence characterization	37
3.3	Results	38
3.3.1	Quality of the designed sequences	39
3.3.2	Finding the GMEC	39
CPU and memory limits for each method	39	
Optimal sequences/structures with up to 10 designed positions	40	
Optimal sequences with 20 or 30 designed positions	42	
3.3.3	Density of states above the GMEC	43
3.4	Conclusions	44
4	PDZ	55
4.1	Introduction	55
4.2	Le modèle d'état déplié	56
4.2.1	Les énergies de référence	56
4.2.2	La vraisemblance des énergies de référence	57
4.2.3	Recherche du maximum de vraisemblance	58
4.3	Méthodes de calcul	60
4.3.1	Fonction énergétique efficace pour l'état replié	60
4.3.2	Les énergies de référence de l'état déplié	61

4.4	Outils d'analyse de séquences	63
4.4.1	Superfamily/SCOP	63
4.4.2	Taux d'identité de séquences	63
4.4.3	Taux d'identité par position	64
4.4.4	Alignements Pfam	64
4.4.5	Score BLOSUM	64
4.4.6	Similarité d'un ensemble à un alignement Pfam	65
4.4.7	Entropie par position	65
4.5	Séquences expérimentales et modèles structurels	66
4.5.1	L'ensemble des protéines PDZ	66
4.5.2	Alignements Blast croisés	66
4.5.3	Sélection des homologues	66
4.5.4	Alignements des protéines expérimentales et leurs homologues	68
4.5.5	Similarité des homologues	68
4.5.6	Les fréquences d'acides aminés	70
4.6	Séquences calculées	83
4.6.1	simulation Monte-Carlo	83
4.6.2	Génération de séquence Rosetta	84
4.6.3	Caractérisation des séquences calculées	84
4.7	Résultats du modèle NEA	85
4.7.1	optimisation du modèle de l'état déplié	85
4.7.2	Tests de reconnaissance de famille	87
4.7.3	Séquences et diversité de séquences	87
4.7.4	Scores de similarité Blosum	88
4.7.5	Tests de validation croisée	89
4.8	Résultats du modèle FDB	91
4.8.1	optimisation du modèle de l'état déplié	91
4.8.2	Tests de reconnaissance de famille	92
4.8.3	Scores de similarité Blosum	92
4.8.4	Taux d'identité à la séquence native	95
4.8.5	Logos des séquences obtenues	95
4.9	Application : Croissance du noyau hydrophobe	96
4.10	Conclusion	101
4.10.1	modèle mis en œuvre	101
4.10.2	tests et application	105

Table des matières

Conclusion	107
Bibliographie	115

Liste des figures

1.1	Principe du l'algorithme du Depth-First Branch and Bound	22
1.2	Exemple de transformation EPT	23
1.3	L'algorithle génétique	25
3.1	width=1cm	52
3.2	width=1cm	52
3.3	Run times for different test calculations and search methods. CPU times per core are shown ; only the REMC calculations use multiple cores (OpenMP parallelization). CFN results are only given for 20 designed positions or fewer. REMC was only done for the larger calculations. Results are shown for a large, representative subset of the test calculations. A few CFN calculations were allowed to run beyond the 24h CPU limit. For clarity, the average Heur and REMC values are included as dashed lines.	52
3.4	Comparison between selected heuristic, MC, and REMC protocols for whole protein design. The best energy obtained with each protocol is shown for the nine test proteins. Zero represents the overall best energy for each protein. The mean “error” is indicated on the right for each protocol (the mean difference from the overall best energy). The protocols are a heuristic method (Heur), an MC method, two REMC protocols with 4 walkers (REMC a, b), and three REMC protocols with 8 walkers (yellow lines and dots, REMC c, d, e). Details of the protocols are in Table 3.1 ; each curve is labelled according to the protocol name.	52
3.5	width=1cm	52
3.6	width=1cm	53
4.1	le cœur PDZ sélectionné	69
4.2	L'alignement de notre sélection de séquences homologues à la protéine NHREF (code PDB :1G9O)	71

Liste des figures

4.3 Une sélection de séquences proteus, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine NHREF (code PDB :1G9O), modèle NEA	72
4.4 L'alignement de notre sélection de séquences homologues à la protéine INAD (code PDB :1IHJ)	73
4.5 Une sélection de séquences proteus, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine INAD (code PDB :1IHJ), modèle NEA	74
4.6 L'alignement de notre sélection de séquences homologues à la protéine Syntenin (code PDB 1R6J)	75
4.7 Une sélection de séquences proteus, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine Syntenin (code PDB :1R6J), modèle NEA	76
4.8 L'alignement de notre sélection de séquences homologues à la protéine GRIP (code PDB : 1N7E)	77
4.9 Une sélection de séquences proteus, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine GRIP (code PDB :1N7E), modèle NEA	78
4.10 L'alignement de notre sélection de séquences homologues à la protéine DLG2 (code PDB 2BYG)	79
4.11 Une sélection de séquences proteus, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine DLG2 (code PDB 2BYG), modèle NEA	80
4.12 L'alignement de notre sélection de séquences homologues à la protéine PSD95 (code PDB 3K82)	81
4.13 Une sélection de séquences proteus, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine PSD95 (code PDB :3K82), modèle NEA	82
4.14 Similarité des séquences des 6 protéines produites par Proteus et Rosetta à l'alignement Pfam RP55, sur l'ensemble des positions	90
4.15 Similarité des séquences des 6 protéines produites par proteus modèle NEA et Rosetta à l'alignement Pfam RP55, sur les positions du cœur hydrophobe.	91
4.16 Similarité des séquences Proteus (NEA et FDB),Rosetta , et des séquences de l'alignement Pfam RP55, sur toutes les positions à gauche et sur les positions du cœur à droite.	96

Liste des figures

4.17 Les séquences conçues par Proteus et les séquences naturelles représentées sous forme de logo pour les positions du cœur hydrophobe	97
4.18 Les séquences conçues par Proteus et les séquences naturelles représentées sous forme de logo pour les positions du cœur hydrophobe	98
4.19 Structure native Tiam1 avec les hydrophobes pour des δ de -0,4,-0,2,0,0,2 et 0,4 sont représentés par un jeu de couleurs allant du rouge foncé au vert clair, en passant par le jaune.	100
4.20 Structure native Cask avec les hydrophobes pour des δ de -0,4,-0,2,0,0,2 et 0,4 sont représentés par un dégradé allant du rouge foncé au vert clair, en passant par le jaune.	102
4.21 Séquences Cask obtenues avec un delta des énergies de références à -0,4,-0,2,0,0,2 et 0,4 et la structure native.Les hydrophobes pour des deltas de -0,4,-0,2,0,0,2 et 0,4 sont représentés par un jeu de couleurs allant du rouge foncé au vert clair, en passant par le jaune.	102

Liste des tables

3.1	Selected MC and REMC protocols	45
3.2	Test proteins	46
3.3	Designed sequence quality measures	46
3.4	Tests with 10 designed positions	47
3.5	Tests with 20 and 30 designed positions	49
3.6	Designed and Pfam sequence entropies	50
4.1	Les groupes d'acides aminés utilisés pour l'optimisation des énergies de référence.	62
4.2	La sélection de domaines protéiques PDZ	66
4.3	E-value et pourcentage d'identité des alignements Blast native versus native pour nos séquences PDZ.	67
4.4	Sélection des homologues.	67
4.5	Similarité des séquences expérimentales homologues, pour les 8 protéines PDZ.	68
4.6	Les énergies de référence obtenues avec l'optimisation 6 protéines.	85
4.7	Les compositions en acide aminé (%) des séquences expérimentales et Proteus après optimisation des E_t^s . Les différences entre expérimentale et théorique sont indiquées entre parenthèses.	86
4.8	Résultats Superfamily pour les séquences Proteus avec le modèle NEA.	87
4.9	Résultats Superfamily pour les séquences Rosetta	88
4.10	Moyenne de l'exponentielle de l'entropie sur les ensembles des positions des protéines	89
4.11	La composition en acide aminé (%) des séquences expérimentales et Proteus après optimisation des énergies de référence. La différence entre expérimentales et Proteus sur les classes est donnée entre crochets.	93
4.12	Les énergies de référence obtenues avec l'optimisation sur 3 protéines.	94

Liste des tables

4.13 Résultats Superfamily pour les séquences Proteus avec le modèle FDB (énergies de références optimisées sur 20 cycles selon les classes + 20 cycles selon les types).	95
4.14 Pourcentage d'identité moyen à la séquence native	95
15 Les tests avec cinq positions actives	110
16 Les tests avec dix positions actives	111
17 Les tests avec vingt positions actives	112
18 Les tests avec trente positions actives	113

Abbreviations

3D tri-dimensionnel(le)	GMEC “Global minimal energie cost”
CASA Coulomb Accessible Surface Area	Pfam ”Protein family databank”
DEE Dead-End Elimination	backbone (ou squelette) chaîne principale de la protéine
CPD Computational Protein Design	$\frac{1}{2}RT$ avec R la constante des gaz parfait et T la température
GB Born Généralisé	
H algorithme heuristique	DFBB Depth-First Branch and Bound
MC algorithme Monte-Carlo	EPT Equivalence Preserving Transformation
NEA Naive Environnement Approximation	
PB Poisson-Boltzmann	MSD Multistart Steepest Descent heuristic
REMC Replica Exchange Monte-Carlo	

Introduction

XXX

Contexte

XXX

XXX

Citation entre crochets [??].

Citation dans le texte ?.

Chapitre 1

le « CPD » : Conception de protéine par ordinateur

L'ingénierie des protéines est l'ensemble des techniques qui ont pour objet de modifier la fonction, ou la structure d'une protéine en modifiant sa séquence d'acide aminés. Les objectifs sont d'augmenter la stabilité des protéines, à modifier des fonctionnements enzymatiques ou encore à ajouter une conformation alternative à une protéine. Dans ce domaine, existe la mutagénèse dirigée dans laquelle une première étape est l'identification des mutations intéressantes pour l'objectif fixé, puis des méthodes de génie génétique sont utilisées pour produire les mutants dont les propriétés souhaitées pourront être vérifiées *a posteriori*. Une deuxième approche est l'évolution dirigée, dans laquelle un ensemble de mutation aléatoire est effectué sur une séquence de protéine d'intérêt et toutes les séquences ainsi produites sont testées afin de trouver la caractéristique attendue. La sélection de fait alors, comme pour l'évolution naturelle dont elle reprend les mécanismes, sur les séquences positives aux tests. Un autre approche est d'utiliser la capacité de calculs des ordinateurs, avec l'apparition des méthodes d'ingénierie des protéines « *in silico* ». L'une d'elles, le « Computational Protein Design » ou CPD conciste à déterminer les séquences d'acides aminés compatibles avec une structure protéique donnée. Ce qui implique la connaissance de la structure dans l'espace 3D de la protéine. Cette méthode comporte trois éléments principaux.

1. la détermination d'un espace de conformation de la protéine C'est sur elle que repose la prédiction de structure des séquences considérées. Elle doit être capable de représenter une ou un petit nombre de conformation de la chaîne principale du polypeptide et tout un ensemble de positions de la chaîne latérale de chacun des résidus.
2. Un fonction d'énergie, qui permet d'évaluer la pertinence des conformations

3. Un algorithme d'exploration de l'espace de conformation qui permet grâce à la fonction d'énergie d'échantillonner les séquences favorables.

Dans ce chapitre, nous aborderons le problème de la modélisation des protéines et de leur espace de conformation et de l'espace conformation-séquence. Puis, nous verrons les fonctions d'énergies classiques pour une conformation. Ensuite, sera détaillé, les approches possibles pour la modélisation du solvant. Plusieurs algorithmes d'explorations de l'ensemble des conformations seront vus. Les principaux programmes CPD seront énumérés. Et enfin, nous présenterons quelques applications du CPD.

1.1 l'espace des séquences-conformations

L'ensemble des cas possibles à prendre en compte, peut se concevoir comme un choix d'un S séquence de longueur N et pour S la détermination d'un conformation C . Ainsi l'espace d'état est celui de l'ensemble de couples. (S,C) pour N donnée. Dans toute la suite on appelle une séquence-conformation un élément de cet ensemble de couples.

L'ensemble des séquences de N acides aminés se conçoit sans difficultés. En revanche, l'ensemble des conformations d'une chaîne polypeptidique doit être défini explicitement.

1.1.1 l'état replié

La mécanique moléculaire propose d'utiliser les représentations de la mécanique classique aux molécules. Les atomes sont représentés par forme de sphère. Ces objets sont alors considérés comme plonger dans un espace 3D euclidien. La protéine dans un milieu aqueux est flexible et en permanence en mouvement. C'est en particulier le cas pour les chaînes latérales ou pour les boucles flexibles. L'espace des états d'une protéine, dans le cadre de la mécanique moléculaire, est alors constitué d'un vaste espace continu de conformation possibles. D'autre part, si un polypeptide a un nombre n de résidus compris entre 50 et 100, et que chacune des n positions de la chaîne peut muter 20 types différents d'acide aminés, le nombre de polypeptides à considérer est égal à n^{20} . Il est donc nécessaire de réduire la taille de l'espace des conformations à prendre en compte. Pour cela, Ponder & Richards [1987] propose une approche en deux points :

1. Le squelette de la protéine est fixé.
2. Les conformations des chaînes latérales sont réduites à un ensemble fini de positions rencontrées dans l'espace 3D.

Ensuite, des variations sur ce principe ont étées introduites, avec notamment l'introductoin de la prise en compte de la mobilité de la chaîne principale dans un ensemble discret ou continu d'état.L'approche qui consiste à générer un ensemble de squelettes et a faire des calculs CPD pour l'ensemble a été utilisé par [137]. Nous présentons maintenant, la modélisation des chaînes latérales et celle du backbone.

les chaînes latérales

les travaux de Finkelstein et Ptitsyn [77],Janin et al [78] ainsi que Ponder et Rischars [87] ont établie que la chaîne latérales des résidues, sur un ensemble de protéines, adope de facon préférentiel un petit ensemble de conformation (fig ??). Janin introduit alors le terme de « rotamère » pour désigner ces conformations. Il est alors possible reduite l'espace continue des conformations des acides aminés à cette ensemble discret de rotamères.la plupart des méthodes de CPD utilisent cette discréttisation.Beaucoup de librairies de rotamères ont été proposées dans la litérarure scientifique. la plupart sont indépendantes du backbone. Mais il existe également des librairies qui dépendent du squelette de la protéine voir [147,148].Le nombre de structures de protéine utilisé est variable , La librairie de Tufféry utlise 53 structures.

Le squelette

Partant du fait que les positionnements des chaînes latérales ne modifie que faiblement la structure adoptée par le backbone,la chaîne principale de la protéine est fixé dans beaucoup de programme CPD. Le problème de la prédiction de struture est alors ramener à celui du placement des chaînes latérale sur ce squelette. De part la configuration particulière de la proline avec le backbone de type est traité a part. Cette approche a obtenue de nombreux succès , voir ?. Cependant, cette approximation peu avoir des consequences importantes.Un type d'acide aminé considéré comme défavorable peut devenir favorable avec une petite adapation du backbone et il a été établie que quelques mouvements de squelettes peuvent faire varié significativement l'énergie de la conformation (Desjarlais et Handel [1999]). En réponse à ce problème, [137] propose de générer un ensemble de squelettes et de faire du CPD pour chacun des exemplaires obtenus. Une autre approche consiste à donner une certaine liberté aux angles α , ψ en introudisant des variations aliéatoires sur ceux-ci (Desjarlais et Handel)[141]. Puis raissament, De Dantas et al [2007] font des simulations avec une minimisation après chaque mouvement de chaîne latérale.Kuhlman et al [2003] optimisent alternativement la structure du squelette et la séquence d'acide aminés.Enfin, citont l'utilisation du classes particulier de mouvement des

squelettes protéiques appelé « backrub ». Ce sont des mouvements naturels du backbone mis en évidence par David et al [2006], à partir de structures cristallographiques. Ces mouvements consiste en des déplacement de l'ensemble $C_\alpha - C_\beta$ à une position i donnée de la chaîne, sans déplacement des carbones $C_{\alpha_{i+1}}$ et $C_{\alpha_{i-1}}$. Ces mouvements backrub ont permis à Georgiev et al [2008] et Smith et Kortemme [2008] d'améliorer la qualité des prédictions des mutants par rapport à des simulations à squelette rigide.

1.1.2 l'état déplié

Lorsque la stabilité d'une protéine est évaluée par une variation de l'énergie libre entre son état déplié et son état replié, il faut connaître l'énergie de l'état déplié. Mais cet état est déstructré et ne correspond pas à une conformation unique ; la modélisation exhaustive est difficile. Une approche simple consiste à représenter cet état par une chaîne étendue ; un résidu de la protéine est en interaction principalement avec le solvant et avec le backbone. Ainsi l'énergie libre de l'état déplié dépend de la séquence uniquement par la composition en acides aminés de celle-ci. En pratique, on peut utiliser pour chaque type d'acide aminé X de la protéine un tripeptide ALA–X–ALA, et on identifie son exposition au solvant à celle de X dans l'état déplié [152] Dahiyat Mayo [1996]. On en déduit une énergie définie par type X que l'on somme pour l'énergie de la protéine dépliée.

1.2 l'énergie de conformation

La fonction d'énergie ou fonction de score, de conformation permet d'évaluer la stabilité (ou une affinité avec un ligand) de chaque conformation de la protéine. Cette fonction doit être capable de prendre en compte les détails des interactions entre les atomes de la protéine, les effets de l'environnement aqueux dans lequel elle se trouve et en même temps être suffisamment rapide pour évaluer en un temps raisonnable une partie la plus significative possible de l'espace des conformations. Une classe importante est constituée des fonctions d'énergie basée sur la mécanique moléculaire.

1.2.1 La mécanique moléculaire

La mécanique moléculaire représente les atomes comme des particules sphériques ayant une charge électrique nette fixe et chaque liaison est modélisée par ressort. Cette mécanique consiste à intégrer les équations du mouvement de la mécanique classique dans un champ de force propre aux molécules étudiées. Ce champ de force décrit les interactions

inter-atomiques du point de vue énergétique et est invariant au cours d'une simulation. Dans la suite, nous appelons E_{MM} l'énergie qui dérive d'un tel champs de force.

Il existe beaucoup de champs de force à la disposition des simulateurs voici les quatre principaux optimisés pour les protéines :

1. AMBER : Assisted Model Building with Energy Refinement [106]
2. CHARMM : Chemistry at HARvard Molecular Mechanics [105]
3. OPLS : Optimized Potential for Liquid Simulations [107]
4. GROMOS :GROningen MOlecular Simulation [108]

L'énergie d'une conformation se définit alors comme la somme de l'énergie E_{MM} et de l'énergie de solvatation :

$$E = E_{MM} + E_{solv} \quad (1.1)$$

Le terme E_{MM} se décompose en :

$$E_{MM} = E_{liées} + E_{non liées} \quad (1.2)$$

avec $E_{liées}$ l'énergie d'interactions des atomes éloignés d'au plus deux liaisons covalentes et E_{inbond} l'énergie des autres interactions. Détaillons ces deux énergies

1.2.2 Les interactions liées

L'énergie d'interaction lié comprend un terme d'elongation des liaisons, un terme de déformation des angles, de rotation des angles dièdres, de torsions, des interactions de Van der Waals, enfin un terme électrostatique de Coulomb.

$$E_{liées} = E_{liaison} + E_{angle} + E_{dièdre} + E_{impropres} \quad (1.3)$$

1. L'énergie de déformations des liaisons $E_{liaison}$ s'exprime de la façon suivante :

$$E_{liaison} = \frac{1}{2} \sum_{i=1}^n k_i (r_i - r_i^0)^2 \quad (1.4)$$

avec l'ensemble des liaisons indexé par i , k_i la force du ressort, r_i la longueur de la liaison et r_i^0 la longueur optimale.

2. L'énergie de déformation des angles E_{angl} est s'exprime :

$$E_{angl} = \frac{1}{2} \sum_{ij} k_{\theta,ij} (\theta_{ij} - \theta_{ij}^0)^2 \quad (1.5)$$

avec θ_{ij} l'angle entre les liaisons i et j, θ_{ij}^0 l'angle optimal et $k_{\theta,ij}$ la force du ressort.

3. L'énergie de déformation des angles dièdres $E_{dièdre}$ est décrite par :

$$E_{dièdre} = \frac{1}{2} \sum_i A_{i,n} [1 + \cos(n\Phi_i - \Phi_0)] \quad (1.6)$$

où n est périodicité de la rotation, $A_{i,n}$ est la constante de torsion, Φ_i l'angle dièdre, c'est à dire pour 4 atomes a_1, a_2, a_3 et a_4 , reliées par 3 liaisons $a_1 - a_2, a_2 - a_3$ et $a_3 - a_4$, l'angle formé par les projets de $a_1 - a_2$ et $a_3 - a_4$ sur un plan perpendiculaire à $a_2 - a_3$. Φ^0 est la phase.

4. L'énergie de déformation des angles impropre E_{impr} exprime la déformation d'un ensemble de 4 atomes par rapport à une conformation planaire ou tétraédrique. Pour un atome a_1 relié à 3 atomes a_1, a_2 et a_3 , elle a la forme :

$$E_{impr} = \frac{1}{2} A (\Phi - \Phi^0)^2 \quad (1.7)$$

Ici, A est la constante de force et Φ représente l'angle entre le plan (a_1, a_2, a_3) et la liaison (a_1, a_2).

voir la figure ?? pour une représentation visuelle.

1.2.3 Les interactions non liées

Les interactions non liées sont les interactions des atomes séparés par plus de trois liaisons covalentes ou qui appartiennent à des molécules différentes. les interactions sont caractérisées par deux termes suivants :

$$E_{nonliées} = E_{elec} + E_{vdw} \quad (1.8)$$

1. L'énergie E_{elec} pour l'énergie électrostatique est donnée par un potentiel de Coulomb de la forme :

$$E_{elec} = \frac{q_i q_j \epsilon}{r_{ij}} \quad (1.9)$$

avec q_i et q_j qui représentent les charges respectives des atomes i et j et r_{ij} représente la distance nette entre les atomes i et j , enfin ϵ et la constante diélectrique du milieu.

2. L'énergie E_{vdW} pour l'énergie de Van der Waals, elle est due aux interactions électriques de faible intensité entre deux atomes, due à la répartition des charges électriques. Cette énergie rassemble des effets des forces de Keesom, Delye, London et

Pauli. Le potentiel de Lennard-Jones est l'approximation classique suivante de cette énergie :

$$E_{vdw} = \sum_{i < j} D_0 [(frac{r_0}{r_{ij}})^{12} - (frac{r_0}{r_{ij}})^6] \quad (1.10)$$

avec D_0 et r_0 des constantes, r_{ij} la distance entre l'atome i et l'atome j . Le premier terme est répulsif

1.2.4 D'autres approches

Même si la mécanique moléculaire prédomine dans le monde du CPD , il existe d'autres approches. Mayo [2006] utilise une fonction empirique pour la modélisation des liaisons hydrogènes, d'une fonction d'énergie coulombienne qui contient un ϵ qui varie en fonction de la distance interatomique. Il existe des fonctions d'énergie comportant des éléments relevant de statisque sur les protéines. Il existe encore des fonctions d'énergie à gros grains notamment dans la modélisation des forces de Van der Walls utiles pour les applications d'interactions protéine-protéine.

1.3 Modélisation du solvant

Les protéines sont étudiées dans un solvant aqueux. C'est à dire que la solution qui est considérée dans les calculs est une mélange dans laquelle l'eau est présente en quantité largement plus importante que la quantité de protéine (les solutés). Bien qu'il existe d'autres types de solvant, ils ne seront pas abordés ici. Les interactions entre solutés et le solvant jouent un rôle clé dans la structure de la protéine mais également dans sa fonction. La modélisation du solvant est alors un point capital dans le CPD. Dès 1933, Bernal et Fowler [127] proposent une modélisation de l'eau liquide au niveau atomique. Depuis, de nombreuses méthodes ont été développées tentant de faire face à la difficulté que cela représente. La connaissance des positions et les vitesses de toutes les molécules d'eau est bien sûr la donnée contenant le maximum d'information dans le cadre de la mécanique moléculaire , mais difficilement gérable compte tenu de la quantité de molécules d'eau qu'il faut considérer.

1.3.1 modèle de solvant explicite

Les modèles qui abordent le problème sous cet angle, c'est à dire celle d'une représentation type mécanique moléculaire dans laquelle l'eau apparaît comme une collection de molécules. Ces molécules s'interagissent au travers d'une énergie potentielle, ou autrement

dit au travers d'un champ de force que décrivent les interactions. Deux composantes de cette énergie potentielle peuvent être considérée :

1. un terme intramoléculaire, qui modélise les interactions entre les atomes d'une même molécule.
2. un terme intermoléculaire, qui modélise les interactions entre les atomes de molécules différentes.

Il est alors assez courant de considérer que l'énergie intramoléculaire n'est jamais modifiée, cela revient à considérer que les longueurs et les angles de liaison des molécules restent fixe au court du temps.

De même pour le terme intermoléculaire, on trouve le plus souvent une modélisation que se limite à considérer uniquement deux types d'énergies :

1. l'interaction de Coulomb, d'énergie potentielle :

$$E_C = \frac{q_1 q_2}{4\pi\epsilon_0 r}$$

avec q_1 q_2 les charges, ϵ_0 la permittivité du vide (1.11)

2. Le potentiel de Lennard-Jones, qui modélise les interactions de Van der Waals, (voir 1.2.3).

Pour avoir la possibilité de calculer les grandeurs d'intérêt du solvant, il faut pouvoir situer le système dans l'espace des phases, c'est à dire l'espace à $6N$ dimensions pour une simulation du solvant avec N molécules d'eau tel que chaque élément de cet espace décrive une configuration des positions et des vitesses de la collection de molécules. Une première méthode est la dynamique moléculaire dans laquelle, à partir d'une configuration initiale des molécules d'eau, les équations de la mécanique classique sont résolues pour déterminer les positions et les vitesses au court du temps. Une seconde méthode consiste à échantillonner l'espace des phases par la méthode Monte-carlo dans laquelle l'espace des phases est visité grâce à une série de déplacements aléatoires qui sont acceptés ou non en fonction de contraintes basées sur l'énergie (voir chapitre ??).

Mais, quelque soit la méthode utilisée, pour pouvoir obtenir une représentation de qualité des effets du solvant sur la protéine, échantillonner correctement les états du solvant dans l'espace des phases, ce qui demande de multiplier les dynamiques moléculaires avec différentes configurations initiales ou de calculer des trajectoires Monte-Carlo suffisamment

longue.D'autrepart, une immersion réaliste du soluté dans de l'eau demande l'utilisation d'un grand nombre de molécules, typiquement plusieurs milliers.

les utilisateurs d' un modèle de solvant explicite se trouvent alors dans la situation où une simulation très couteuse et qui en plus consomme la plus grande partie de la puissance de calcul à évaluer les interactions des molécules d'eau entre-elles.Il apparaît alors naturellement la nécessité d'utiliser les méthodes numériques moins couteuses et plus efficaces.

1.3.2 modèle implicite

Le principe des modèles implicites du solvant est de tenter de représenter l'effet moyen du solvant sur la protéine par l'utilisation d'un milieu continu dans lequel la protéine serait immergée.

l'effet du solvant sur la protéine peut être vu comme la différence entre l'énergie libre de la solution solvant/soluté avec l'énergie libre d'un système dans lequel le solvant et le soluté sont séparés. Notons cette différence ΔG_{solv} , elle est appelé l'énergie de solvatation.

On peut décrire ΔG_{solv} comme la somme de trois termes :

$$\Delta G_{solv} = \Delta G_{solv}^{elec} + \Delta G_{solv}^{VdW} + \Delta G_{solv}^{cav} \quad (1.12)$$

avec :

1. ΔG_{elec} l'effet électrostatique, qui correspond à la reorganisation des charges de la protéine dans le solvant
2. ΔG_{VdW} est l'effet des forces de Van der Waals.
3. ΔG_{cav} est l'effet qui correspond au coût de la création de la cavité pour le solvant, en terme d'entropie et de pression.

Il est d'usage de réunir les deux derniers termes en une contribution non polaire , dites hydrophobe.Il possible de d'approximer ce terme hydrophobe en utilisant la surface accessible au solvant de la protéine (voir figure ??) :

$$\Delta G_{solv}^{VdW} + \Delta G_{solv}^{cav} \approx \Delta G_{solv}^{SA} = \sum_i \sigma_{t_i} A_i \quad (1.13)$$

avec A_i la surface accessible au solvant de l'atome i et σ_{t_i} un facteur pour chaque type atomique t_i ajusté pour retrouver les énergies de solvatation obtenue par expérience ou autres.Ce facteur figure la propension de chaque type d'atomes à être enfoui ou exposé au solvant (Wesson Eisenberg [1992]).

1.3.3 modèle CASA

La présence des molécules d'eau qui sont très fortement polarisé induit une atténuation des interactions électrostatiques entre atomes de la protéine. On parle de l'écrantage induit par l'eau.

Le modèle « Coulomb Accessible Surface Area » (CASA) réduit l'effet électrostatique du solvant à l'écrantage subit par la protéine.

$$\Delta G_{solv} \approx E_{screen} + \sum_i \sigma_{t_i} A_i \quad (1.14)$$

avec

$$E_{screen} = \left(\frac{1}{\epsilon} - 1\right) E_{coul} \quad (1.15)$$

Ainsi, l'écrantage est décrit par un facteur unique pour toutes les interactions électrostatiques. La simplicité de ce modèle proposé par Wesson et Eisenberg [212] fait de lui un modèle fréquemment utilisé. Cependant, il est en difficulté pour le traitement de la surface des protéines.

1.3.4 modèle Poisson Boltzmann

La méthode de Poisson Boltzmann est très utilisée parce qu'elle fournit des résultats de grande précision, prend en compte l'effet ionique et toute la forme de la protéine. Dans cette méthode, la protéine est supposée fixe, elle forme une cavité dans un milieu continu diélectrique qui peut être polarisé (on parle de continuum). Il s'agit donc encore d'un modèle de solvant implicite. Par contre les charges de la protéine sont traitées de façon explicite. Ainsi ce modèle est capable de prendre en compte l'écrantage du solvant sur les interactions intrasolvent, mais aussi les interactions électrostatiques entre groupes chargés de la protéine et solvant polarisé. Le continuum est munie d'une distribution de charge ρ_P , alors le potentiel électrostatique dans ce milieu est donné par l'équation de Poisson :

$$-\nabla \cdot \epsilon(x) \nabla \phi(x) = \rho(x) \quad (1.16)$$

avec $\epsilon(x)$ le coefficient diélectrique qui dépend du milieu, $\phi(x)$ le potentiel électrostatique. Typiquement ϵ pr

L'équation de Poisson-Boltzmann est une expansion de l'équation de Poisson dans laquelle les charges d'ions mobiles sont pris en compte. La théorie de Debays-Hückel donne la distribution des ions comme suivant une loi de Boltzmann (d'où son nom) :

$$\rho_i = C_i \exp(-q_i(\psi(x) + V_i(x))) \quad (1.17)$$

avec C_i une constante pour chaque type d'ion i , q_i la charge partielle en dans le potentiel $\psi(x)$, et $V_i(x)$ une fonction d'énergie stérique représente la difficulté des charges mobiles à pénétrer dans le soluté.

La distributions de charge devient :

$$\rho_{PB} = -\nabla \cdot \epsilon(x) \nabla \phi(x) + C_i \exp -q_i(\psi(x) + V_i(x)) \quad (1.18)$$

Malheureusement, il n'existe pas , pour les protéines , de solution analytique à ??q. Il faut effectué une résolution numérique. Plusieurs programmes sont à la disposition de la communauté, DelPhi[], APBS[] qui sont basés sur la méthodes des différences finies [] et des éléments finis []].

Une fois le potentiel électrostatique calculé, l'énergie libre électrostatique est donné par :

$$\Delta G_{elec} = \frac{1}{2} \int \rho \phi dx \quad (1.19)$$

Quelque soit l'approche utilisée les calculs sont couteux pour une protéine entière.

1.3.5 modèle Born Généralisé

Le modèle de solvant GB pour « Generalized-Born » se base sur une expression nouvelle de l'équation de Poisson-Boltzmann ; ce qui ouvre la voie à une nouvelle classe d'approximations.L'objectif est alors de donner les résultats de mêmes qualités que PB avec un coût numérique bien inférieur.On procede de la façon suivante :Dans un premier temps, on cherche à évaluer l'énergie libre électrostatique d'un ensemble de N particules, de rayon de Born R_i de charge q_i dans une milieux de constante diélectrique ϵ :

$$\Delta G_{elec} = -\frac{1}{2} \pi (1 - \frac{1}{\epsilon}) \sum_{i=1}^N q_i^2 \alpha_i \quad (1.20)$$

$$E_{elec} = E_{Coul} + \Delta G_{solv} \quad (1.21)$$

$$E_{elec} = \sum_{i \neq j}^N q_i q_j r_{ij} + \Delta G_{solv} \quad (1.22)$$

chez Najette

$$\Delta G_{solv} = \sum_i \Delta E_i^{self} + \sum_{i < j} \Delta E_{ij}^{int} \quad (1.23)$$

Dans article Fran

$$\Delta G_{solv} = \sum_{ij} g_{ij} \quad (1.24)$$

$$g_{ij} = g(r_i, r_j) = \tau q_i q_j (r_{ij} + b_i b_j \exp(-r_{ij}/4b_i b_j))^{-1/2} \quad (1.25)$$

avec $\tau = \frac{1}{\epsilon_w - \frac{1}{\epsilon_p}}$, ϵ_w étant la constante diélectrique du solvant, ϵ_p étant la constante diélectrique de la protéine. b_i et b_j sont les rayons de solvatations.

Cette expression est exacte lorsque les rayons de Born ne se recouvrent pas. Mais les rayons de Born des différentes particules ne sont pas connus. Dans le cas où les rayons de Born se recouvrent, Il a été proposée une généralisation de la formule ?? [23] :

$$\Delta G_{elec} = -18\pi(1 - \frac{1}{\epsilon}) \sum_{i=1}^N \sum_{j>i}^N \frac{q_i q_j r_{ij}^2}{4\alpha_i \alpha_j \exp(-r_{ij}^2/4\alpha_i \alpha_j)} \quad (1.26)$$

Le calcul des rayons de Born α_i se fait par le calcul de l'énergie libre électrostatique de l'atome i , dans la situation où toutes les autres particules ont une charge ramenée à zéro. Ainsi, chaque charge de la protéine est caractérisée par sa distance au solvant.

1.4 l'algorithme d'exploration

Après, un choix de l'espace d'état des conformations/séquences, après la construction d'une fonction d'énergie qui qualifie les états d'intérêt. Il reste, pour compléter les ingrédients de bases du CPD, à choisir un algorithme d'exploration de l'espace d'état qui permet la sélection d'un sous-ensemble de séquences selon la pertinence établie par la fonction de score. Dans l'idéal, l'objectif du CPD est d'obtenir l'ensemble de séquences d'acide aminé qui sont compatibles à une structure 3D de repliement ou qui réalisent une fonction donnée. Un tel algorithme a pour grand défi de faire face à l'immensité de l'espace d'état. On peut classer les différentes approches utilisées en deux groupes.

Le premier groupe est constitué des méthodes déterministes dans lesquelles les choix effectués sont toujours soit déterminé a priori soit déterminé en fonction des éléments obtenus au cours de l'exécution du programme qui l'implémente. On trouve par exemple dans ce groupe, les méthodes exhaustives qui peuvent être appliquées à de petits espaces conformationnels ou encore les méthodes dites semi-exhaustives dans lesquels la complexité combinatoire est réduite en autorisant uniquement certaines conformations à certains moments de l'exploration. Une classe intéressante de ce groupe est constituée des algorithmes exacts qui se focalisent non pas sur

l'objectif du CPD , mais sur l'obtention de la conformation de meilleure énergie on parle alors de « Global Minimum Energy Cost » (GMEC). Typiquement, ces algorithmes exploitent les structures de la fonction d'énergie et de l'espace d'état.Si elle est additifs par paires de rotamères,un type particulier de méthodes est alors exploitable.Bien souvent, ces algorithmes nécessitent un pré-calcu des énergies et le stockage de celles-ci , ce qui peut être problématique , par exemple dans les situations où le backbone est reflexe, un nombre d'état possible de la chaîne latérale vient multiplié par autant la taille de l'espace de phase. Or parmi les états qu'il faut calculer le nombre de ceux qui ne sont sans intérêt pour l'exploration de l'espace peut devenir considérable. Ce qui constitut un facteur de ralentissement .

Le second groupe est celui des méthodes stochastiques et/ou heuristiques. Elles ont vocation à déterminer des solutions de qualité sans obtenir de garantie sur l'optimum en énergie des solutions, dans un temps d'exécution réaliste. Elles sont non-déterministes c'est à dire qu'elles choisissent certains éléments de façon aléatoire, en pratique la « source de hasard » est constituée par des générateurs de nombre pseudo-aléatoire.Ellles ont l'avantage d'être utilisables sur les espaces d'états non-discretisé, par exemple [154], les rotamère peuvent varier de façon continue.Elle ne nécessite pas de connaissance a priori sur l'espace des phases.il est souvent possible de mettre en place un système simple qui stocke un jeu d'énergie après une première utilisation, pour le réutiliser lors de futurs nécessités. Par contre la convergence bien qu'elle puisse être établie en théorie, reste en pratique difficile à cerner et n'offre pas la possibilité de reconnaître le GMEC.Ce qui conduit au besoin de choisir une condition d'arrêt de l'exécution sans liant direct avec ce minimum.

Pour rendre possible l'utilisation de plusieurs outils algorithmique , il devient nécessaire d'imposer des contraintes sur la structure de la fonction d' l'énergie. Il existe alors en CPD doit être décomposable en une énergie dite propre qui rassemble les effets de la chaîne latérale et du backbone, d'une part et une énergie d'interaction rotamère-rotamère qui se décompose comme une somme sur des interactions sur les couples de chaînes latérales prises deux à deux.

on a alors la formule suivante :

$$Ec = \sum_i E_i(r_i) + \sum_i E_{ij}(r_i, r_j) \quad (1.27)$$

Nous présentons quelques algorithmes parmis les plus utilisés.

1.4.1 Algorithme du champ moyen

Le principe de la méthode du champ moyen est le substituer l'ensembles des interactions d'un rotamère r_0 avec les autres rotamères par une interaction unique moyenne. Pour calculer une interaction moyenne, la probabilité de Boltzmann est utilisée de la façon suivante, on note $P(r_i)$ la probabilité que la chaîne latérale à la position i soit dans l'état rotamère r_i , on a :

$$P(r_i) = \frac{1}{N_i} \exp(-\beta E(l_i)) \quad (1.28)$$

avec N_i le nombre total de rotamère à la position i , et $\beta = \frac{k_B T}{R}$, R étant la constante des gaz parfaits et T la température.

Si l'on ne limite au cas où une énergie d'interaction pour un résidu est la somme des interactions avec les autres positions de la chaîne, alors l'énergie d'interaction moyenne en i est la somme des interactions impliquant le rotamère r_i pondéré par le poids de Boltzmann de l'autre rotamère, c'est à dire

$$E(r_i) = \sum_{j \neq i} \sum_l^N E(r_i, l_j) P(l_j) \quad (1.29)$$

avec l_j parcourant tous les rotamères aux positions autres que i .

Les formules () et (), constituent un système itératif. Ainsi, l'algorithme débute

1. Etape 0 à chaque position, les rotamères sont équiprobable.
2. Etape 1 Les énergies moyennes sont calculées grâce à la formule () .
3. Etape 2 De nouvelles probabilités de Boltzmann sont calculées à partir des énergies moyennes précédentes

Les étapes 1 et 2 sont répétées jusqu'à convergence des énergies. Cette méthode garantit la convergence vers un ensemble de rotamère un plus stable à chaque position placé dans son « environnement moyen ». Le temps de calculs avec cet algorithme augmente de façon linéaire avec le nombre de résidus de la protéine. Ce qui en fait un des algorithmes les plus rapides.

1.4.2 Dead-End Elimination

Le « Dead End Elimination » (DEE) consiste à éliminer des mauvais rotamères , ou des mauvaises combinaisons de rotamères qui ne peuvent pas aboutir à la combianisons de rotamère qui minimisent de façon global, l'énergie. Comme pour le champ moyen, l'énergie doit être décomposable en une énergie propre et une énergie de paires.

1.4. L'algorithme d'exploration

Il y a deux critères d'élimination (Goldstein [1994]),(Desmet [1992]) :

1. Le critère simple : Le rotamère r_i du résidu i est éliminé si la meilleure énergie (la plus faible) qu'il est possible d'obtenir pour une conformation comprenant ce rotamère est moins bonne que la pire énergie obtenue avec un rotamère r'_i à la même position. Ainsi, une expression mathématique peut être :
avec $E(c)$, l'énergie d'une séquence-conformation c de l'espace d'état.

$$si \min_{r_i \in C} E(C) > \max_{r'_i \in C} E(C), alors$$

r_i est éliminé. (1.30)

En utilisant l'expression d'une fonction d'énergie décomposée par paires :

$$E(c) = \sum_i E_i(r_i) + \sum_{i \neq j} E_{ij}(r_i, r_j) \quad (1.31)$$

Le critère peut s'écrire :

$$si E_i(r_i) + \sum_{j \neq i} \min_{r_j} E_{ij}(r_i, r_j) > E_i(r'_i) + \sum_{j \neq i} \max_{r_j} E_{ij}(r'_i, r_j), alors$$

r_i est éliminé. (1.32)

avec, N le nombre de positions .

2. Le critère double : Il exprime une condition analogue sur un couple de rotamère (r_i, r_j) pour pouvoir l'éliminer, dans le sens qu'il n'est pas possible qu'il fasse partie du GMEC..Pour son expression on introduit l'énergie d'une paire (r_i, r_j) :

$$E^{r_i, r_j} = E_i(r_i) + E_j(r_j) + E_{ij}(r_i, r_j) \quad (1.33)$$

On a alors,

$$si E^{r_i, r_j} + \sum_{k=1}^N \min_{r_k} (E_{ik}(r_i, r_k) + E_{jk}(r_j, r_k)) > E^{r'_i, r'_j} + \sum_{k=1}^N \min_{r_k} (E_{ik}(r'_i, r_k) + E_{jk}(r'_j, r_k)), alors la paire (r_i, r_j) est éliminée.$$

r_i, r_j est éliminé. (1.34)

L'élimination d'un couple (r_i, r_j) , n'exclue pas pour autant la présence de r_i ou de r_j dans la solution optimale.

3. le critère de Goldstein ?? Il s'agit d'une amélioration du critère simple du DEE. Il peut exister des situations dans lesquelles l'énergie avec un rotamère r_i est toujours diminuée en la remplaçant par un autre rotamère r'_i . On peut donc l'éliminer. Or, cela n'implique pas forcément la condition du critère simple. Ce critère peut s'exprimer de la façon suivante :

$$si E_i^{r_i} - E_i^{r_j} + \sum_{k=1}^N \min_{r_k} (E_{ik}(r_i, r_k) + E_{jk}(r_j, r_k)) > 0, alors$$

r_i est éliminé. (1.35)

Voir la figure

Au cours de l'optimisation, les deux critères peuvent être utilisé alternativement, jusqu'à qu'il n'y est plus de rotamère à éliminer. Il est alors possible d'utiliser une approche hexausicve sur l'espace d'états réduit obtenu, pour obtenir le GMEC.

Cette méthode permet dans les bons cas (par exemple pour les petits systèmes) de converger en un temps raisonnable[26]. Beaucoup de variantes du DEE [158,152] ont été proposé Pierce NA, Spriet JA, Desmet J, Mayo SL. (2000). Conformational splitting : a more powerful criterion for dead-end elimination. J Comput Chem 21 : 999-1009. Notamment en travaillant sur des ensembles de plus de deux rotamère.

figure figure...

1.4.3 le CFN

La méthode des réseaux de fonction de coût est issue du domaine de l'optimisation combinatoire. Il s'agit ici encore d'une méthode utilisable si la fonction d'énergie est décomposable par paires. De plus elle nécessite un pré-calcul des énergies. Il s'agit avant tout d'une recherche de la séquence-conformation qui minimise l'énergie globale, mais qui dans certaines conditions peut également fournir un ensemble de séquence-conformation proche du GMEC.

Un réseau de fonction de coût , ou cost function network (CFN) , est défini par un triplet X,D,C tel que :

1. X un ensemble de variables à valeurs dans N
2. D, un ensemble de domaines pour les variables de X
3. C, un ensemble de fonctions, dont chaque élément porte sur un sous-ensemble S de X et donne un coût strictement positif pour chaque combinaison de valeurs des variables de S.

Un problème CPD peut alors être représenté sous la forme d'un CFN, en associant chaque résidu i variable d'une protéine par une variable v_i du CFN, L'ensemble des rotamères possible en i définissant le domaine de v_i . Le terme E_i représentant l'énergie 'self' correspond à E_i et le terme E_{ij} à une fonction de C avec $S=v_i, v_j$. A une séquence-conformation correspond donc un valeur de D pour chaque variable de X, on parle de solution du CFN. La recherche du minimum globale d'énergie revient alors à trouver une solution qui minimise la somme de toutes les fonctions de coût.

La résolution est tenté généralement par un algorithme de type « Depth-First Branch and Bound » (ou DFBB). les ingrédients sont les suivants :

1. un principe de séparation

Un ensemble des solutions peut être vu comme un sommet S_0 d'une arborescence sans branche. La séparation est l'action de partager, selon certain critère, ce sommet initial en sous-ensembles qui devient les sommets fils de S_0 , ce partage devant constituer une partition de l'ensemble des solutions de départ. Le critère de séparation classique est d'énumérer les valeurs possibles d'une variable v_i du CFN. A chacune des valeurs x de v_i on définit le sous-ensemble de S_0 ayant dans toutes ses solutions, v_i égale à x . voir exemple en ???. Ainsi, si v_i à N valeurs possibles, N fils de S_0 sont créés.

2. un majorant Le majorant du problème correspond au meilleur coût connu. Il majore le GMEC.
3. un minorant d'un sommet Le minorant correspond à une valeur que l'on sait être inférieur ou égal au coût de toute les solutions d'un sommet
4. un principe d'évaluation L'évaluation d'un sommet S , c'est déterminer un de ces minorants . Donc si un sommet est évalué et est supérieur au majorant courant, on sait qu'aucune solution de S ne peut être le GMEC. La totalité du sous arbre peut être élagué (i.e exclu de l'optimisation).
5. une stratégie de développement La stratégie de développement consiste à choisir une méthode de développement de l'arbre des solutions ; C'est à dire déterminer l'ordre sur les sommets de l'arborescence dans lequel on va appliquer le critère

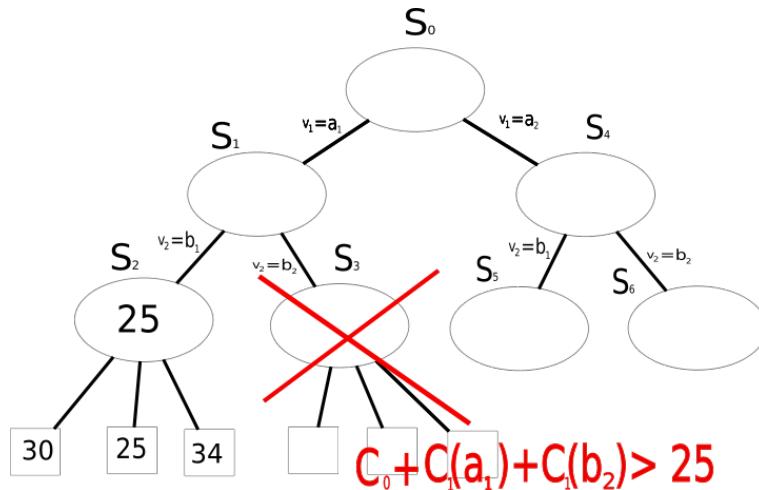


Figure 1.1 – Principe du l'algorithme du Depth-First Branch and Bound

de séparation. Dans le DFBB, la stratégie consiste à descendre dans les branches jusqu'à trouver un sous-arbre qu'il est possible d'élagué, alors l'algorithme remonte d'une branche pour redescendre dans une autre direction. Ce parcours en profondeur à l'avantage de limiter l'utilisation de la mémoire, parce qu'il n'est nécessaire de conserver que la description de la branche qui a été exploitée.

Il est alors nécessaire de trouver de bons minorants.

Equivalence Preserving Transformation

L'idée principale des cohérences locales est transformer le CFN en un CFN dans lequel le coût des affectations complètes sont identique (elles sont appelées EPT pour « Equivalence Preserving Transformation »), de façon à faire apparaître de bons minorants Schiex,2000. Le principe est de déplacer les coûts entre fonctions dans le but de les attribuer aux fonctions qui portent sur une seule variable au plus (les coûts unitaires et les coûts constants) .l'avantage ici, est de pouvoir conserver les transformations dans un chemin de l'arbre de recherche tout au long de l'optimisation.Le principe est présenté sur un exemple inspiré de la thèse de Seydou Traoré.la figure ??.. représente 6 CFN équivalents.Il existe deux types de transformation la première appelé projection consiste à transférer un cout de l'ensemble des couts unaires vers le cout constant C_0 , c'est la transition AB ,des coûts binaires vers le coût constant C_0 , (transition BC) ou encore des coûts binaires vers les coûts unaires (transition DE). Le second type de transformation fait l'inverse, c'est à dire transfert un cout d'une fonction unaire vers les fonction binares impliquant la même valeur de variable transition CD .Une telle transformation permettant une augmentation du transfert total vers C_0 , CFN F.

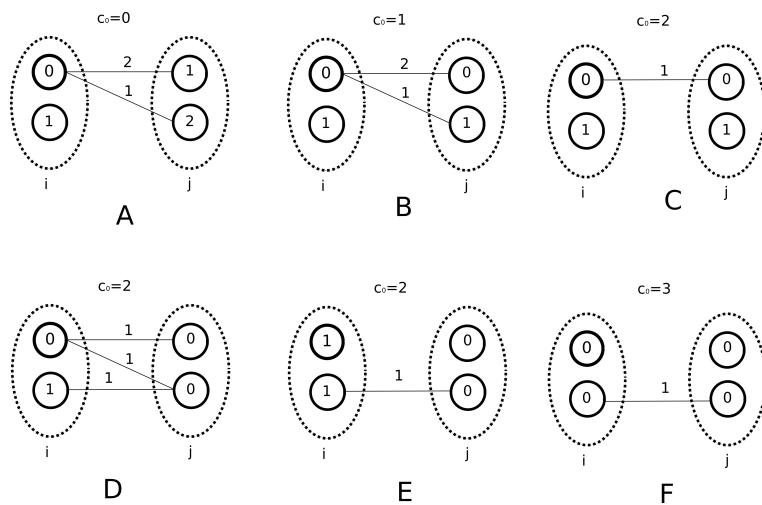


Figure 1.2 – Exemple de transformation EPT

Cet approche a montré sa supériorité dans le domaine du CPD , avec l'outil toulbar2 par rapport à tout un ensemble de résolveur de CFN parmi les plus réputé actuellement qui exploitent notamment l'approche DEE/A* ou le backtracking ?? . toulbar2 en combinant FFBB,EPT et DEE a était capable de trouver le GMEC dans le plus grand nombre de problèmes proposés aux différents outils.

1.4.4 l'heuristique Multistart Steepest Descent (MSD)

En 2000,M Wernisch, Hery et Wodak partent du constat que l'espace des phases et les fonctions d'énergies qu'il est possible d'utiliser ne capture que partiellement la réalité des protéines in vivo. Ainsi, d'une part, le GMEC ne correspond par forcement à la séquence la plus stable.D'autre part, il est bien connu que des protéines homologues éloigné avec des taux d'identité à moins de 50% peuvent conserver quasiment la même structure 3D, ce qui révèle l'immensité des séquences-conformations compatibles à un pli.Ainsi, l'objectif n'est plus de raisoudre une problème d'optimisation, mais d'exhiber une ensemble de séquences de basses énergie. Ils proposent alors une heuristique concut pour le CPD. Il s'agit d'un procedure simple qui a pour objectif de produire très facilement une grande quantité de séquences-conformations de basses énergie,sans se focaliser sur l'optimum.

Un cycle heuristique se déroule de la façon suivante : Au départ, une séquence-conformation est construite en attribuant de façon aléatoire un type d'acide aminé et un rotamère à chaque position de la chaîne. Ensuite, En parcourant la séquence du début jusqu'à la fin,l'algorithme procede par des ajustements successifs à chaque position.Pour chaque position i de la séquence, toutes les états possibles sont évalué, le reste de la

séquence et des rotamères étant fixé. Le meilleur rotamère est alors fixé en i . La séquence est ainsi, ajuster par plusieurs passage successif, jusqu'à convergence de l'énergie, voir (??).

Un cycle est très rapide, ce qui permet de produire dans les cas usuels, plusieurs milliers de séquences par heures. L'utilisation mémoire hormis la gestion du stockage des énergies est quasi-nulle. Il n'impose pas que la fonction d'énergie soit pairwise, mais il s'en accorde très bien, en rendant possible l'utilisation de la mémoire par bloc des énergies impliquant une position i donnée.

```
1 pour chaque cycle heuristique faire
2   choisir une séquence-conformation  $S$  aléatoirement ;
3   tant que l'énergie de  $S$  est améliorée faire
4     pour  $i$  allant de la première position de  $S$  jusqu'à la dernière faire
5       fixer  $S$  sauf à la position  $i$  ;
6       déterminer le meilleur rotamère possible en  $i$  ;
7       fixer  $S$  en  $i$  avec ce rotamère ;
8   sauvegarder  $S$  ;
```

1.4.5 Algorithme génétique

Holland et ses collaborateurs introduisent une nouvelle approche inspirée des principes biologique de la sélection naturelles, avec des opérations comme les mutations, les croisements et la sélection. L'algorithme génétique a pour objectif d'obtenir un ensemble de solution proche de l'optimum en un temps raisonnable. Le schéma générale du déroulement est le suivant. Une population de séquences-conformations est générée de façon aléatoire. L'énergie de tous les membres de la population est évaluée. Un critère de sélection basée sur l'énergie est appliqué à la population, qui donc diminue. Un ensemble de mutations aléatoires et un ensemble de croisement sont appliqués sur la nouvelle population. Donc la population augmente. Alors, une condition d'arrêt est évaluée, si elle n'est pas réalisée l'algorithme retourne à l'étape d'évaluation.

Le membre de meilleure énergie de la population tend à se reproduire le plus vite dans la population. Donc la valeur moyenne de l'énergie de l'ensemble des séquences-conformations converge. On peut voir ici, que le nombre de membres de la population est un paramètre de l'algorithme. Plus ce nombre est faible plus la convergence va être rapide. A contrario, plus ce nombre est grand, plus l'exploration de l'espace d'état est meilleure. Les atouts de l'algorithme génétique sont sa capacité à franchir des barrières énergétiques par des

changements de séquences rapides via le mécanisme des croisements et sa capacité à optimiser en parallèle différents secteurs de la structure.

1.4.6 Monte-Carlo

Dans son acceptation la plus générale, un algorithme Monte-Carlo est un algorithme stochastique (il utilise une source de hasard) qui donne approche probablement la solution exacte en un temps d'exécution déterminable a priori.Cela le distingue, parmi les algorithmes stochastiques , d'un algorithme de Las Vegas qui donne un résultat exact dans un temps d'exécution non déterministe , et d'un algorithme D'Atlantic City qui donne des résultats probablement correct dans un temps probablement rapide. Parmi les algorithmes Monte-Carlo, les algorithmes Monte-Carlo par Chaînes de Markov (MCMC) ont été particulièrement étudiés. Ce ne sont pas des algorithmes d'optimisation, mais des algorithmes d'échantillonnage d'une distribution de probabilité π : *On peut générer des éléments x_i de l'espace des phases tels que la distribution que constitue $D = \{x_i, i \in T\}$ converge vers π .*

Il existe de nombreuses méthodes pour répondre à cette question, algorithme de Newton-Cotes ?? , algorithme du rejet ?? les résultats sont bons lorsque la dimension est relativement petite ?? tandis que les MCMC sont bien adapté dans les situations où l'espace de états est de grande dimension.Un autre avantage des algorithmes MCMC est qu'elles ne nécessité pas la connaissance de la constante de normalisation de π .*Cela constitue un attrait important parce qu'en dehors de la situation où cette constante de normalisation est connue, il n'y a pas de moyen pratique de la déterminer.*

Dans la suite, nous donnons des conditions suffisantes pour que ce type l'algorithme converge dans le cadre d'un espace d'état E fini, puis nous detaillons le plus célèbre d'entre-eux, l'algorithme de Metropolis-Hasting.

L'idée générale,est de générer un élément x_i en utilisant dans les états déjà visités uniquement x_{i-1} .*Cela signifie que l'élément x_i est choisi au hasard parmi les états que l'on a visité jusqu'à présent.*

On appelle processus stochastique, une suite de variable aléatoire X_{nn_0} à valeurs dans E .*Une chaîne de Markov est un processus stochastique tel que les étapes sont indépendantes conditionnées à l'état initial et aux étapes précédentes.*

$$\forall n \geq 0, \forall (i_0, i_1, \dots, i_{n-1}, i) \in E^{n+1}, P(X_n = i | X_{n-1} = i_0, X_{n-2} = i_1, \dots, X_1 = i_{n-2}, X_0 = i_{n-1}) = P(X_n = i | X_{n-1} = i_0) \quad (1.36)$$

Une chaîne de Markov est homogène si :

$$\forall n \geq 0, \forall (i, j) \in E^2, P(X_n = i | X_{n-1} = j) = P(X_{n+1} = i | X_n = j) \quad (1.37)$$

C'est à dire que la probabilité de transition P est indépendante de n .Dans la suite, on ne considère que des chaînes de Markov homogènes.Et on note $P(X_n = b | X_{n-1} = a) = P(b|a)$

dans la suite, pour simplifier les notations, on utilise l'écriture matricielle avec $\mu_0 = (p(X_0 = i))_{i \in E}$, on a alors

$\mu_1 = \mu_0.P$, avec . le produit matriciel. et

$$\mu_k = \mu_0.P^k$$

Le problème revient alors à trouver une application de probabilité de transition $P^{k-} > \pi$

Pour cela introduisons deux conditions supplémentaires :

Le principe de la balance détaillée :

Le principe de la balance détaillée est un principe générale de comportement des systèmes dynamiques, il a été utilisé par Boltzmann pour la construction de la physique statistique et Cercignani et Lampis ont montré qu'il était vrai pour les gaz polyatomiques. Il peut s'exprimer de la façon suivante :

Pour un système dynamique à l'équilibre S , a et b deux éléments de l'espace des phases, la probabilité de transition de a vers b est égale à la probabilité de transition de b vers a , ou encore :

$$\forall i, j \in S \mathcal{P}(i \rightarrow j) = \mathcal{P}(j \rightarrow i) \quad (1.38)$$

Pour pouvoir appliquer ce principe aux chaînes de Markov, introduisons une définition proche :

Soit q une probabilité sur E , une chaîne de Markov est dite reversible par rapport à q si

$$\forall i, j \in E, q(j)P(i|j) = q(j)P(j|i) \quad (1.39)$$

Dans ce cas, on a alors :

$$\forall i \in E, q.P(i) = \sum_{j \in E} q(j)P(i|j) = \sum_{j \in E} q(i)P(j|i) = q(i) \sum_{i \in E} P(j|i) = q(i) \quad (1.40)$$

Ainsi, la probabilité q avant la transition P vaut la probabilité après, la chaîne est dite stationnaire pour q . Il faut alors construire une chaîne de Markov reversible pour la cible π .

Mais cela ne suffit pas, en effet même si une distribution stationnaire est connue, rien ne dit que la chaîne va entrer dedans.

Si E est discret, une chaîne de Markov reversible est dites ergodique si et seulement si :

1. pour tous i, j de l'espace d'état il existe un chemin de i vers j de probabilité non nulle.
2. Il existe aucun x de E , tel que la chaîne revient en x périodiquement.

1.4. l'algorithme d'exploration

3. Au court du temps,tout les états sont visités par la chaîne avec une probabilités non nulle.

On a alors, le résultat suivant : Une chaîne ergodique converge vers son unique distribution stationnaire.

l'algorithme de Metropolis

Nous pouvons maintenant décrire l'algorithme de Metropolis-Hastings, L'idée initiale de Metropolis a été de construire un algorithme qui échantille la distribution de Boltzmann, qui donne la probabilité d'un état x_i dans le système en fonction de son énergie et de la température :

$$\pi^B(i) = \frac{1}{Z} \sum_{j \in E} e^{-E_j/kT} \quad (1.41)$$

avec E_{x_i} l'énergie de x_i , T la température et k la constante de Boltzmann. La constante de normalisation de la probabilité s'appelle la fonction de partition du système. Elle est particulièrement difficile à calculer.

On définit, alors la probabilité de transition P comme le produit de deux probabilités :

$$P(j|i) = sel(j|i)acc(j|i) \quad (1.42)$$

avec sel une probabilité de sélectionner l'état j de E lorsque le système est dans l'état i et acc la probabilité d'accepter l'état j étant en i. Si l'on souhaite une chaîne respectant le principe de la balance détaillée par rapport à π^B , on a :

$$\pi^B(i)sel(j|i)acc(j|i) = \pi^B(j)sel(i|j)acc(i|j) \quad (1.43)$$

Si on se limite à une probabilité de sélection symétrique :

$$\pi^B(i)acc(j|i) = \pi^B(j)acc(i|j) \quad (1.44)$$

ce que équivaut à

$$acc(j|i)acc(i|j) = \frac{\pi^B(j)}{\pi^B(i)} = \frac{e^{-E_j/kT}}{e^{-E_i/kT}} = e^{(\delta E)/kT} \quad (1.45)$$

avec $\delta E = E_j - E_i$.

Ainsi, le calcul de la fonction de partition est évité.

Métropolis propose alors la probabilité d'acceptation suivante :

si

$\delta E > 0$ alors $\text{acc}(i|j) = \exp(\delta E/kT)$ si non $\text{acc}(i|j) = 1(1.46)$

On voit que , si $\delta E > 0$ alors $\text{acc}(j|i)=1$, et donc on a bien :

$$\forall i,j \in E, \text{frac}(\text{acc}(j|i)\text{acc}(i|j)) = \exp(\delta E/kT)$$

Il suffit alors de choisir une probabilité de selection symétrique et ne violant pas ?? pour obtenir notre convergence.

Par la suite Hastings, généralise l'algorithme à une probabilité *sel()* non symétrique avec *acc()* telle que :

si

$\delta E > 0$ alors $\text{acc}(i|j) = \exp(\delta E/kT)\text{sel}(j|i)\text{sel}(i|j)$ si non $\text{acc}(i|j) = 1(1.47)$

Si l'on injecte cette nouvelle probabilité dans ??, on a encore le respect de la balance détaillée et la convergence.

```

1 Une séquence-conformation  $S_0$  est choisie aléatoirement;
2 pour chacun des pas idélatrajectoire faire
3   choisir une proposition  $S'_i$  à partir d'une probabilité conditionnelle  $\text{sel}(.,S)$ ;
4   calculer une probabilité d'acceptation  $\text{acc}=...$ ;
5   si  $\text{acc} \geq 1$  alors
6      $S_{i+1} = S'_i$  ;
7     sauvegarder  $S_{i+1}$  ;
8   sinon
9     alors  $S_i + 1 = S'_i$  avec une probabilité  $\text{acc}$  ;
10    sauvegarder  $S_{i+1}$  ;
11    sinon
12       $S_{i+1} = S_i$ 
13

```

Dans toute la suite, on désigne l'algorithme de Metropolis-Hastings comme l'algorithme Monte-Carlo (MC).

Monte-Carlo avec échanges de répliques (REMC)

Une amélioration de l'algorithme de Métropolis-Hastings, connu sous le nom de « Replica Exchange Monte-Carlo » a été introduite par Swendsen and Wang (Swendsen RH and Wang JS (1986) Replica Monte Carlo simulation of spin glasses Physical Review Letters 57 : 2607-2609). L'objectif est d'accélérer la convergence en permettant au processus stochastique de visiter plusieurs zones énergétiques simultanément. Ainsi, l'algorithme couple l'exploration dans les basins de basses énergies , importante pour la recherche des solutions optimum avec, l'exploration dans des zones de plus hautes énergies, ce qui facilite la sortie des minimum locaux.

On considère N simulations MC du système à différentes températures. On associe La chaîne de Markov sur l'espace d'état E^N , constituée des N -uplets (x_1, \dots, x_n) avec les x_i éléments de E , et $1 < i < N$ indexant les températures. On ajoute alors un nouveau type de déplacement , celui qui consiste à intervertir au temps t , l'état x_t^i de la simulation à la température i avec l'état x_t^{i+1} à la température $i + 1$. On a alors :

$$X_{t+1} = (x_{t+1}^1, \dots, x_{t+1}^i, x_{t+1}^{i+1}, \dots, x_{t+1}^n) = (x_t^1, \dots, x_t^i, x_t^{i+1}, \dots, x_t^n) \quad (1.48)$$

Pour que X respecte la balance détaillée il est nécessaire d'utiliser ce nouveau mouvement dans certains conditions.

Pour

- 1 Lancement en parallèle de N marcheurs MC aux températures ordonnées (t_1, \dots, t_n) ;
- 2 **pour** *les pas p multiples d'une constante P faire*
- 3 choisir aléatoirement i compris entre 1 et $N - 1$, ce qui sélectionne les marcheurs aux températures t_i et t_{i+1} ;
- 4 la probabilité d'acceptation suivante est calculée acc=... ;
- 5 **si** $acc \geq 1$ **alors**
- 6 Les marcheurs échangent leur température ;
- 7 **sinon**
- 8 Les marcheurs échangent leur température , avec la probabilité acc ;

1.5 Les programmes CPD

1.6 Les succès

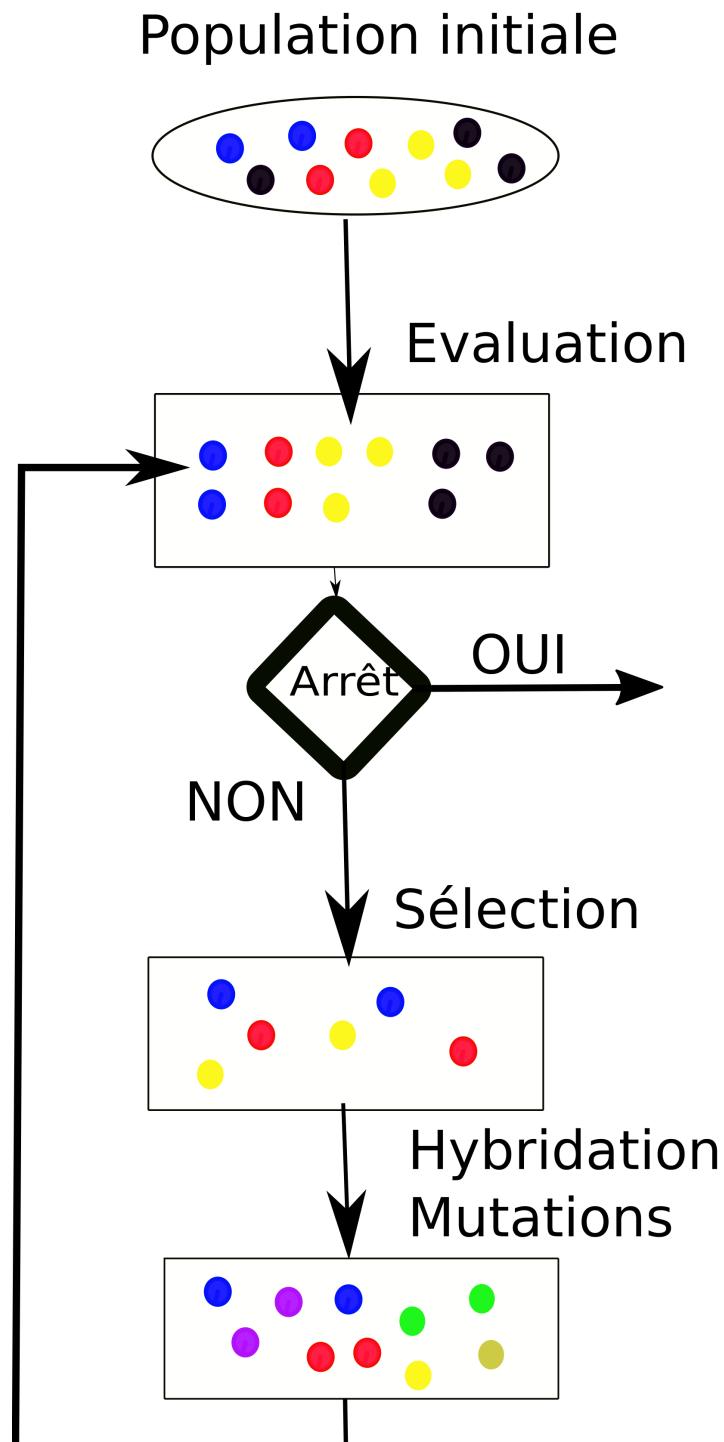


Figure 1.3 – L'algorithle génétique

Chapitre 2

proteus

Chapitre 3

Comparing stochastic search algorithms for computational protein design

David Mignon and Thomas Simonson*

†Laboratoire de Biochimie (UMR CNRS 7654), Dept. of Biology, Ecole Polytechnique, Palaiseau, France *Corresponding author. Email : thomas.simonson@polytechnique.fr

Abstract

Computational protein design depends critically on an energy function, a conformation space model, and an algorithm to search the space of sequences and conformations. We compare three search algorithms that are stochastic : a heuristic method, a Monte Carlo method (MC), and a Replica Exchange Monte Carlo method (REMC) that propagates several walkers, or replicas at different temperatures. The methods are applied to nine test proteins, with 1, 5, 10, 20, 30, or all amino acid positions allowed to mutate, for a total of about 200 tests. Results are also compared to a recent, exact, “Cost Function Network” method (CFN) that identifies the global minimum energy conformation (GMEC) in favorable cases. The designed sequences accurately reproduce the experimental amino acid types for positions in each protein’s hydrophobic core. The heuristic and REMC methods are in good mutual agreement, yielding very similar optimal solutions ; they accurately reproduce the GMEC when it is known, with a few exceptions. Plain MC performs well for most but not all cases, occasionally departing from the GMEC by 3–4 kcal/mol. With REMC, the diversity of the sequences sampled, as measured by the mean sequence entropy, agrees well with exact enumeration in the range where the latter is possible : about 2 kcal/mol above the GMEC. Beyond this range, room temperature

Chapitre 3. Comparing stochastic search algorithms for computational protein design

walkers routinely sample sequences up to 10 kcal/mol above the GMEC, providing thermal averages and at least a rough solution to the inverse protein folding problem.

3.1 Introduction

Computational protein design (CPD) has developed into an important tool for biotechnology ??????. Starting from a 3D structural model, CPD explores a large space of possible sequences and conformations, to identify protein variants that have certain predefined properties, such as stability or ligand binding. Conformational space is usually defined by a library of sidechain rotamers, which can be discrete or continuous, and by a finite set of backbone conformations or a specific repertoire of allowed backbone deformations. The energy function usually combines physical and empirical terms ????. Both solvent and the unfolded protein state are described implicitly.

The number of amino acid positions that are allowed to mutate can vary, depending on the problem of interest, from 2 or 3 to several dozen. Thus, the combinatorial complexity can be enormous, so that speed is important, as well as accuracy. In addition, it is usually important to identify not one but several high-scoring sequences, for at least three reasons. First, if the typical error in the energy function is σ_E , we should enumerate all the possible sequences/structures within one or two σ_E of the optimal one. Second, it may be of interest to characterize the diversity of a sequence family, by enumerating sets of sequences compatible with its backbone fold (the “inverse folding problem”) ?????. Third, we may want to compute properties that are averaged over structural and possibly sequence fluctuations at a given temperature, which requires that we explore solutions within the thermal range. An example is the calculation of ligand binding constants, following a method introduced recently ?. Calculation of acid/base constants by constant-pH Monte Carlo is another example, which can also be seen as a subproblem of CPD, where sidechain protonation state changes are treated as mutations ???.

Thus, the complexity and cost of a CPD calculation will depend on several factors. While energy calculations usually represent the bulk of the cost, the power and efficiency of the exploration method are also very important. Several exploration methods exist that can identify exactly the global minimum energy sequence and conformation, or GMEC. These include “dead end elimination” methods, or DEE ??, branch-and-bound methods ??, and cost function network methods ?. While some of these methods can handle large problems, they usually cannot enumerate suboptimal solutions within a large interval σ_E above the GMEC (more than a few kcal/mol). Partly for this reason, stochastic methods remain popular, such as Monte Carlo (MC) ?. MC has two advantages : with an appropriate setup, it samples sequences/conformations from a known, Boltzmann distribution, and it can be readily combined with enhanced sampling methods developed in the broader field of molecular simulations, such as Replica Exchange or umbrella sampling ??.

Chapitre 3. Comparing stochastic search algorithms for computational protein design

Our goal here is to assess three stochastic exploration methods for a series of CPD problems of increasing complexity. The first method is a heuristic method that is not guaranteed to find the global minimum energy conformation, or GMEC, but has been effective in applications ??. The second method is a Monte Carlo (MC) exploration, which samples sequences/conformations from a Boltzmann distribution ??. The third is an enhanced, multi-walker MC, which performs “replica exchange” ??. Several walkers, or replicas are propagated at different temperatures, and exchange conformations at regular intervals according to a MC test. We refer to it as REMC. These methods are also compared to a fourth method that is exact, in the sense that it can provably identify the GMEC in favorable cases ?. It is based on ‘cost function networks’, or CFN, where the cost function is the energy, and the network refers to the set of interacting amino acids. The CFN method uses a depth-first branch-and-bound search through a tree of rotamer assignments, with fast integer arithmetic for the energy evaluations. It can also enumerate all the sequence/conformation combinations within a given energy range δE (not too large) above the GMEC. It is implemented in the Toulbar2 program, by Schiex and coworkers. Other exact methods exist, some of which appear to be even faster than CFN. Our goal, however, is not to “rank” the stochastic and exact methods, but rather to compare our three stochastic methods to each other, and this is facilitated if an exact enumeration of low energy states has also been done.

We use a CPD model that is rather simple but representative of a large class of applications. We use a discrete set of sidechain rotamers, a fixed backbone structure, and we assume that the energy function is pairwise additive; i.e., the energy has the form of a sum over residue pairs ??. With these simplifications, all possible residue pair interactions can be computed ahead of time and stored in a lookup table ?; exploration is then done in a second stage. Thus, the cost of energy calculations and sequence/structure exploration are well-separated. The method is implemented in the Proteus CPD package ?? (except for the CFN sequence exploration, done with Toulbar2). Our MC framework is presented in some detail below; the other methods are recalled more briefly. < We considered nine test proteins from three structural classes : SH3, SH2, and PDZ domains. For each one, we chose different numbers and sets of residues to mutate and we applied the different exploration methods, using several possible parameterizations for each one. To characterize the different methods, we compared their speed, their ability to identify the GMEC, and their sampling of suboptimal sequences/conformations above the GMEC. The designed sequences were characterized by computing their similarity to natural sequences, their classification by the Superfamily fold recognition tool ??, and their sequence entropies. For the few cases where there were large differences between the methods (several kcal/mol

between best-scoring sequences), the 3D structural models were compared. Overall, the heuristic method is the most successful in identifying low energy solutions, while REMC is almost as successful but has the advantage of sampling from a Boltzmann distribution over a large energy range, yielding thermal averages.

3.2 Methods

3.2.1 Monte Carlo : general framework

We consider a polypeptide of n amino acids. Its sequence S is written $S = t_1 t_2 \cdots t_n$, where t_i is the sidechain type of amino acid i . We assume that each amino acid i can take on a few different types t, t', \dots that form a set T_i . For each sequence, there are two classes of structures : folded and unfolded. For the folded form, all the sequences S share the same, precise geometry for the polypeptide backbone ; only the sidechain positions can vary. Specifically, the sidechain of each amino acid i can explore a few discrete conformations or “rotamers” r, r', \dots (around 10 per type t_i). The structure of the unfolded form is not specified ; the energy is assumed to be independent of the particular unfolded structure, and to have the additive form :

$$E_u(S) = \sum_{i=1}^n E_u(t_i) = \sum_{i=1}^n (e_u(t_i) - kT \log n_u(t_i)), \quad (3.1)$$

where $E_u(t_i)$ is a free energy associated with sidechain type t_i in the unfolded state, and the rightmost term separates it into an energy component $e_u(t_i)$ and a conformational entropy term, where kT is the thermal energy and $n_u(t_i)$ is the number of conformations or rotamers available to sidechain type t_i in the unfolded state.

We perform a Monte Carlo simulation ??? where one copy of the folded protein is explicitly represented. The unfolded state is included implicitly, by propagating the simulation with the energy function $E_M = E_f - E_u$ (the folding energy). One possible elementary MC move is to change a rotamer r_i in the current folded sequence ; the energy change is $\Delta E_M = \Delta E_f = E(\dots t_i, r'_i \dots) - E(\dots t_i, r_i \dots)$. Another possible move is a mutation : we modify the sidechain type $t_i \rightarrow t'_i$ at a chosen position i in the folded protein, assigning a particular rotamer r'_i to the new sidechain. The energy change is

$$\Delta E_M = \Delta E_f - \Delta E_u = (E_f(\dots t'_i, r'_i \dots) - E_f(\dots t_i, r_i \dots)) - (E_u(t'_i) - E_u(t_i)) \quad (3.2)$$

ΔE_M measures the stability change due to the mutation (for the given set of rotamers) ; it is as if we performed the reverse mutation $t'_i \rightarrow t_i$ in the unfolded form.

Chapitre 3. Comparing stochastic search algorithms for computational protein design

If the moves are generated and accepted with an appropriate Metropolis-like scheme, the Markov chain will visit states according to their Boltzmann probability :

$$p_M(S,c) = \frac{e^{-\beta(E_f(S,c)-E_u(S))}}{\sum_{S'} \sum_{c'} e^{-\beta(E_f(S',c')-E_u(S'))}} \quad (3.3)$$

where $\beta = 1/kT$ and the subscript M indicates probabilities sampled by the Markov chain. For two conformations c, c' of sequence S, the Markov probability ratio is $p_M(S,c)/p_M(S,c') = e^{-\beta(E_f(S,c)-E_f(S,c'))}$. For two sequences S, S', the probability ratio is

$$\frac{p_M(S)}{p_M(S')} = \frac{\sum_c e^{-\beta(E_f(S,c)-E_u(S))}}{\sum_{c'} e^{-\beta(E_f(S',c')-E_u(S'))}} = \frac{e^{-\beta\Delta G_{\text{fold}}(S)}}{e^{-\beta\Delta G_{\text{fold}}(S')}} \quad (3.4)$$

In the ratio of Markov probabilities, we recognize the ratio of Boltzmann factors for S and S' folding, so that we have the second equality, where $\Delta G_{\text{fold}}(S)$ denotes the folding free energy of sequence S (respectively, S').

Eq. (3.4) has a simple interpretation : the Markov chain, with the chosen energy function $E_M = E_f - E_u$ and appropriate move probabilities, leads to the same distribution of states as a macroscopic, equilibrium, physical system where all sequences S, S', ... are present at equal concentrations, and are distributed between their folded and unfolded states according to their relative stabilities. This is exactly the experimental system we want our Markov chain to mimic. In this interpretation, a Monte Carlo mutation move $S \rightarrow S'$ amounts to unfolding one copy of S and refolding one copy of S'.

It remains to specify the move generation probabilities and choose an appropriate acceptance scheme ????. Let $\alpha(o \rightarrow n)$ be the probability to select a trial move between two states o and n and $acc(o \rightarrow n)$ the probability to accept it. If the simulation obeys detailed balance, we have

$$N(o)\pi(o \rightarrow n) = N(n)\pi(n \rightarrow o), \quad (3.5)$$

where $N(o), N(n)$ are the equilibrium populations of states o and n. With “ergodic” move sets such as the one used here (see below), detailed balance is guaranteed in the limit of a very long simulation. To produce Boltzmann statistics, we choose the acceptance probabilities ??? :

$$acc(o \rightarrow n) = \exp(-\beta\Delta E_M) \frac{\alpha(n \rightarrow o)}{\alpha(o \rightarrow n)} \quad \text{if } \Delta E_M > 0; 1 \text{ otherwise} \quad (3.6)$$

where $\Delta E_M = E_M(n) - E_M(o)$ is the o \rightarrow n energy difference.

For a rotamer move at a particular position in the polypeptide chain, of type t , we define the move generation probability as $\alpha(o \rightarrow n) = \frac{1}{n_f(t)} = \alpha(o \rightarrow n)$; all possible choices for the new rotamer are equiprobable, forward and backward rotamer moves have the same generation probability, and Eq. (3.6) reduces to the simple Metropolis test ?.

For a mutation move at a particular position, we define $\alpha(o \rightarrow n)$ as follows :

- (a) select a new type t' with equal probabilities $\alpha_t(o \rightarrow n) = \frac{1}{N}$ for all N possible types ;
- (b) choose a rotamer r' for the new sidechain with equal probabilities $\alpha_r(o \rightarrow n) = \frac{1}{n_f(t')}$ for all $n_f(t')$ possible folded state rotamers.

The overall probability is therefore

$$\alpha(o \rightarrow n) = \alpha_t(o \rightarrow n)\alpha_r(o \rightarrow n) = \frac{1}{Nn_f(t')} \quad (3.7)$$

The $o \rightarrow n$ and $n \rightarrow o$ probabilities are different whenever the old and new sidechain types have different numbers of possible rotamers. With these move probabilities, the mutation acceptance probability can be written :

$$acc(t \rightarrow t') = e^{-\beta(\Delta E_f - \Delta E_u)} \frac{n_f(t)}{n_f(t')} = e^{-\beta(\Delta E_f - \Delta e_u)} \frac{n_f(t)n_u(t')}{n_u(t)n_f(t')} \quad \text{if } \Delta E_M > 0 \quad (3.8)$$

$$= 1 \quad \text{otherwise} \quad (3.9)$$

If the number of rotamers in the folded and unfolded states are the same, $n_u = n_f$, the fraction on the right will cancel out. However, the rotamer numbers also appear in the energy change that determines whether the move is uphill, ΔE_M .

With REMC, several simulations (“replicas” or “walkers”) are propagated in parallel, at different temperatures ; periodic swaps are attempted between two walkers’s conformations. The swap is accepted with probability

$$acc(t \rightarrow t') = \text{Min} \left[1, e^{(\beta_i - \beta_j)(\Delta E_i - \Delta E_j)} \right] \quad (3.10)$$

where β_i , β_j are the inverse temperatures of the two walkers and ΔE_i , ΔE_j are their folding energies ??.

3.2.2 MC and REMC : implementation details

For plain MC, we use one- and two-position moves, where either rotamers, types, or both are changed. For two-position moves, the second position is selected among those that have a significant interaction energy with the first one, 10 kcal/mol or more. For REMC, we use four or eight walkers, with thermal energies kT_i that range from 0.125 to 2 or 3

kcal/mol, and are spaced in a geometric progression : $T_{i+1}/T_i = \text{constant}$, following Kofke ?. Conformation swaps are attempted at regular intervals, between walkers at adjacent temperatures. The precise parameter settings are given in Table 3.1.

3.2.3 Heuristic sequence optimization

The heuristic sequence optimization uses an iterative minimization ??. One “heuristic cycle” proceeds as follows : an initial amino acid sequence and set of sidechain rotamers are chosen randomly. These are improved in a stepwise way. At a given amino acid position i , the best amino acid type and rotamer are selected, with the rest of the sequence held fixed. The same is done for the following position $i + 1$, and so on, performing multiple passes over the amino acid sequence until the energy no longer improves or a set, large number of passes is reached (500 passes). The final sequence, rotamer set, and energy are output, ending the cycle. The method can be viewed as a steepest descent minimization, starting from a random sequence, and leading to a nearby, local, (folding) energy minimum. Below, we typically perform ~ 100.000 heuristic cycles for each protein, thus sampling a large number of local minima on the energy surface.

3.2.4 Cost function network method

The CFN method is implemented in the Toulbar2 program ??. The Proteus energy matrices are converted to the Toulbar format with a perl script. With this format, all the interaction energies are approximated as positive integers, without loss of generality. We used Toulbar2 version 0.9.7.0 with a recommended parameterization (options -l=3 -m -d : -s) ; for the unsuccessful cases (GMEC not identified) we systematically repeated calculations with version 0.9.6.0 and a more aggressive protocol (options -l=1 -dee=1 -m -d : -s). To enumerate sequence/conformation pairs that have energies higher than the GMEC, Toulbar2 is run with the “suboptimal” option and an energy threshold. Available memory was limited to 30 gigabytes.

3.2.5 Energy function

The energy function has the form :

$$E = E_{\text{bonds}} + E_{\text{angles}} + E_{\text{dihe}} + E_{\text{impr}} + E_{\text{vdw}} + E_{\text{Coul}} + E_{\text{solv}} \quad (3.11)$$

The first six terms represent the protein internal energy. They were taken from the Charmm19 empirical energy function ?. The last term represents the contribution of

solvent. We used a “Coulomb + Accessible Surface Area”, or CASA implicit solvent model ?? :

$$E_{\text{solv}} = \left(\frac{1}{\epsilon} - 1 \right) E_{\text{Coul}} + \sum_i \sigma_i A_i \quad (3.12)$$

Here, ϵ is a dielectric constant that scales the Coulomb energy set 23, following earlier tests ?. A_i is the solvent accessible surface area of atom i ; σ_i is a parameter that reflects each atom’s preference to be exposed or hidden from solvent. The solute atoms were divided into 4 groups with the following σ_i values (cal/mol/Å²) : unpolar (-5), aromatic (-40), polar (-8) and ionic (-10). Hydrogen atoms were assigned a surface coefficient of zero. Surface areas were computed by the Lee and Richards algorithm ?, using a 1.5 Å probe radius. Pairwise additivity errors for the surface energy term were corrected by applying a reduction factor of 0.5 to buried pairs ?. Energy calculations were done with a modified version of the Xplor program ??.

The energies $E_u(t)$ associated with the unfolded state were determined empirically to give reasonable amino acid compositions for the protein families considered here ?; they are reported in Supplementary Material.

3.2.6 Test systems and preparation

We considered nine protein domains, from the SH3, SH2, and PDZ families, listed in Table 3.2. Each domain is known to fold stably and has an associated crystal structure used for our calculations. Systems were prepared and energy matrices computed using procedures described previously ?. Briefly, each PDB structure was minimized through 200 steps of conjugate gradient minimization. For each residue pair, interaction energies were computed after 15 steps of energy minimization, with the backbone fixed and only the interactions of the pair with each other and the backbone included. Sidechain rotamers were described by a slightly expanded version of the library of Tuffery et al ?, which has a total of 228 rotamers (sum over all amino acid types).

3.2.7 Sequence characterization

Designed sequences were compared to the Pfam alignment for the corresponding family, using the Blosum40 scoring matrix and a gap penalty of -6. Each Pfam sequence was also compared to its own Pfam alignment. For these Pfam/Pfam comparisons, if a test protein T was part of the Pfam alignment, the T/T self comparison was left out, to be more consistent with the designed/Pfam comparisons. If the test protein T was not part of the Pfam alignment, we used Blast to identify its closest Pfam homologue H

Chapitre 3. Comparing stochastic search algorithms for computational protein design

and left the T/H comparison out, for consistency. The Pfam alignments were either the “seed” alignment for each family (around 50 sequences) or much larger, “full” alignments, with 6287, 3052, and 14944 sequences, respectively, for the SH3, SH2, and PDZ families. Similarities were computed for protein core residues, defined by their near-complete burial, and listed in Results.

Designed sequences were submitted to the Superfamily library of Hidden Markov Models ??, which attempts to classify sequences according to the SCOP classification ?. Classification was based on SCOP version 1.75 and version 3.5 of the Superfamily tools. Superfamily executes the hmmscan program, which implements a Hidden Markov model for each SCOP family and superfamily ; here hmmscan was executed with an E-value threshold of 10^{-10} , using a total of 15438 models to represent the SCOP database.

To compare the diversity in the designed sequences with the diversity in natural sequences, we used a standard, position-dependent sequence entropy ?, computed as follows :

$$S_i = - \sum_{i=1}^6 f_j(i) \ln f_j(i) \quad (3.13)$$

where $f_j(i)$ is the frequency of residue type j at position i , either in the designed sequences or in the natural sequences (organized into a multiple alignment). Instead of the usual, 20 amino acid types, we employ six residue types, corresponding to the following groups : {LVIMC}, {FYW}, {G}, {ASTP}, {EDNQ}, and {KRH}. This classification was obtained by a cluster analysis of the BLOSUM62 matrix ?, and also by analyzing residue-residue contact energies in proteins ?. To get a sense of how many amino acid types appear at a specific position i , we usually report the residue entropy in its exponentiated form, $\exp(S_i)$, which ranges from 1 to 6.

3.3 Results

Our main focus is to characterize sequence/structure exploration methods and their ability to sample low energy sequences. We begin, however, by showing that the sequences we sample are similar to experimental ones. Indeed, the performance of exploration methods depends on the shape and ruggedness of the energy surface, and should be tested in situations where the energy function is sufficiently realistic, as judged by the quality of the designed sequences. After that, we compare the ability of the four exploration methods to identify low energy sequences, including the GMEC. Finally, we consider the diversity of sequence sets, or density of states sampled by each method.

3.3.1 Quality of the designed sequences

We first report information on the quality of our designed sequences. We use sets of REMC sequences to illustrate the main features. The best REMC protocol, REMCD (Table 3.1) was used. Results with the other exploration methods are expected to be similar. Indeed, while the methods sometimes exhibit differences of up to a few kcal/mol between their best sequences, the average sequence quality of the 100–1000 best sequences is typically similar between methods. Table 3.3 summarizes results for our 9 test proteins in design calculations where all positions were allowed to change types. All mutations were allowed, except mutations to/from Gly and Pro, since these are likely to change the backbone structure. REMC was done with 8 replicas at temperatures between 0.175 and $3 kT$ units. Simulation lengths were 750 million steps (per replica). The top 10000 sequence/conformation combinations were retained, corresponding to 200–400 unique sequences. For the 1A81 SH3 domain, none of these sequences was recognized by Superfamily as an SH3 family, or even superfamily member. For the other 8 proteins, all the retained sequences were recognized as members of the correct superfamily *and* family, with match lengths and E-values given in Table 3.3. Thus, our designed sequences are largely similar to experimental ones. Sequence identities to wildtype (including 1A81) are 31% on average (Table 3.3), similar to earlier studies with the same energy function ???. Representative sequence logos for the protein core are shown in Fig. 1, illustrating the agreement between designed and experimental sequences. Similarity scores were computed between the designed sequences and experimental sequences from the Pfam database ?. For the protein core region, the similarity is similar to that between experimental sequences, as shown in Fig. 2.

3.3.2 Finding the GMEC

CPU and memory limits for each method

The ability of an exploration method to sample low energy sequences depends on the CPU and memory resources available, as well as on detailed parameterization choices. Here, we set somewhat arbitrary limits, to remain within a practical run situation. For CFN, we set a maximum time limit of 24 hours and a memory limit of 30 gbytes. For the heuristic method, we used 110,000 heuristic cycles, increased to 330,000 or 990,000 cycles in a few cases; even for these cases, run times did not exceed 24 hours. For MC, we ran up to 10^9 simulation steps, which corresponded to CPU times of 9 hours at most. For REMC, we ran $0.75 \cdot 10^9$ simulation steps per replica, with a few exceptions. We used an OpenMP, shared memory parallelization on a single processor, with one replica per core.

Chapitre 3. Comparing stochastic search algorithms for computational protein design

Total CPU time per core was never more than 3 hours, for a total CPU use of less than 24 hours. For the heuristic, MC, and REMC methods, memory requirements are modest ; about 2 gbytes for the largest calculations. Run times are shown in Fig. 3 as a function of the number of designed positions, which varies from one position to the entire protein (about 90 positions). For comparison, the CPU time needed to compute the energy matrix for a single pair of designed positions, using an advanced energy function (Generalized Born solvent plus a sophisticated surface area term) and a single core of a recent Intel processor is about 5 hours.

The MC and REMC methods require choices of move probabilities and temperatures, which affect the sampling in ways that vary from protein to protein. Fig. 4 shows the lowest energy sampled with a small collection of protocols : one heuristic, one MC, and five REMC protocols, applied to our nine proteins, with all positions allowed to mutate (except Gly/Pro). The protocol details are given in Table 3.1. For these large design problems, the GMEC is not known. Instead, for each protein, the overall best energy (the best of the seven protocols) is taken as the reference, or zero value. For a given protein, the best energy varies by up to 12 kcal/mol from one protocol to another (compare the 1BM2 REMCa and REMCc energies or the 1CKA MC and REMCd energies). For each protocol, the best energy obtained was averaged over the nine proteins, giving a mean “error” ; these values are also reported in Fig. 4. The lowest mean error is 1.0 kcal/mol, with REMCd. In other words, REMCd gives a best energy that is, on average, 1.0 kcal/mol above the overall best energy. Based on these and other similar tests, for the rest of this work, we used the specific MC protocol in Table 3.1 and the REMCd replica exchange protocol, which are generally good but not necessarily optimal for every situation.

Optimal sequences/structures with up to 10 designed positions

As our first series of tests, we did calculations for each test protein with zero, one, or five designed positions. Results are summarized in Fig. 5. With zero positions, only rotamers are optimized (at all positions in the protein). With one, we systematically designed each position of each protein in turn (plus rotamers at all positions). With 5, we picked the positions randomly, close together in the structure, in 5 different ways, for a total of 45 tests. Two positions were considered close by if they have at least one rotamer combination that gives an interaction energy of 10 kcal/mol or more (in absolute magnitude ; eg, steric overlap). In all these cases, CFN found the GMEC very rapidly (seconds) ; the heuristic also found the GMEC, with much longer run times (an hour). MC found the GMEC in all but a few cases (Fig. 5), with run times of a few minutes.

3.3. Results

As a second series of tests, we chose randomly for each protein a set of ten positions to design ; the other positions had fixed types but explored all possible rotamers. The selected positions were close by in the protein structure. For each protein, we made five separate choices of positions to design, for a total of 45 test cases. The CFN, heuristic, and MC methods were run for all 45 cases ; REMC was run only when MC gave a poor result (6 cases, involving 5 proteins). Results are summarized in Fig. 5 and Table 3.4. 20 cases where all methods found the GMEC are not listed in the Table, leaving 25 where at least one method did not find the GMEC. CFN performed very well : only in one case did it not find the GMEC. The lowest energy was sampled in this case with the heuristic, and the best CFN energy was 5.7 kcal/mol higher (despite using the more aggressive CFN protocol).

The heuristic performed about as well as CFN for 10-position design. In one case, CFN did not find the GMEC and the heuristic gave the lowest energy (2BYG-1). In 39 cases, the heuristic found the GMEC. In 3 cases, it was within 0.15 kcal/mol of the GMEC, with no mutations (only rotamer differences). In one case (1CKA-5), it was 0.29 kcal/mol above the GMEC, with no mutations. Tripling the number of heuristic cycles allowed the GMEC to be reached (within 0.07 kcal/mol) in all these cases, with run times below 6 hours. There was only one real failure, 1M61-2, where the best heuristic solution was 3.5 kcal/mol above the GMEC, with 3 mutations relative to the GMEC. For this case, the GMEC was recovered (within 0.01 kcal/mol) if the number of cycles was increased to 990,000, for a run time of 7 hours. Switching from the heuristic structure (after 330,000 cycles) to the GMEC requires concerted changes in 3 adjacent sidechain positions. This is only possible during a heuristic cycle if there is a downhill, connecting pathway made of single position changes, which is evidently very rare for this particular test. Thus, the heuristic method can only find the GMEC if it draws the right combination of types/rotamers at the very beginning of a cycle ; hence the need for 990,000 cycles.

Plain MC did slightly less well for 10-position design. In 21 cases, it found the GMEC. In 18 cases, its best sequence was within 0.2 kcal/mol of the GMEC, with 0–3 mutations (one on average). Notice that 0.2 kcal/mol is the thermal energy for the MC protocol employed. In 6 cases, its best sequence was between 0.9 and 4.5 kcal/mol above the GMEC, with 2–7 mutations (3 on average). For these 6 cases, REMC was run, and sampled sequences within 0.40 kcal/mol of the GMEC, except for one case where its best sequence was 0.80 kcal/mol above the GMEC. Overall, MC or REMC reached the GMEC to within 0.40 kcal/mol in all but one case. A 0.40 kcal/mol energy difference is actually less than the average pairwise additivity errors in the energy function ???, and so one might consider this performance to be about as good as the CFN and heuristic methods. In terms of

Chapitre 3. Comparing stochastic search algorithms for computational protein design

speed, for 10-position design, all the methods were comparable (a few hours per run on average).

Optimal sequences with 20 or 30 designed positions

We did similar tests with 20 designed positions, selected randomly in 5 different ways for each protein, as above. Results are given in Fig. 5 and Table 3.5. CFN found the GMEC in 28 out of 45 cases ; in 2 others, it found the best energy of the 4 methods. For 6 of these 30 cases, the more aggressive protocol was necessary, and run times were 2–22 hours (11 on average). For 14 of the other 15 cases, the best CFN energy was 0.1–7.5 kcal/mol above the best solution found by the other methods, and 2.8 kcal/mol on average, despite using the more aggressive protocol. For the worst case, the CFN energy was 13.9 kcal/mol above the best solution.

The heuristic method found the GMEC in 22 of the 28 cases where it is known. For the other 6 cases, it was within 0.40 kcal/mol of the GMEC, with 0–4 mutations (2.7 on average). For the 17 cases where the GMEC was not identified by CFN, the heuristic produced the lowest energy of all methods, except one case (104C-1) where it was 0.35 kcal/mol above CFN. Overall, the heuristic either found the best energy of the four methods or was within 0.40 kcal/mol of the best energy.

MC converged to the best energy in 11 cases ; in 25 other cases, it was within 0.50 kcal/mol of the best energy. In the other 9 cases, its best energy was at most 3.2 kcal/mol above the best energy (sampled by the heuristic and/or CFN). Finally, REMC was done for all the test cases. In 6 cases, its best energy was more than 0.50 kcal/mol from the best energy. However, the differences were notably smaller than for plain MC, with an average of just 0.8 kcal/mol for the 6 worst cases and a maximum (for 1G90-1) of 1.25 kcal/mol.

The same tests were done with 30 designed positions ; see Fig. 5 and Table 3.5. CFN found the GMEC in just one case ; in 5 others, it did not find the GMEC but gave the lowest energy overall. In 4 other cases, it was within 0.50 kcal/mol of the best energy sampled by the other methods. For the other 35 cases, its best energy was higher than the best method, with differences of 10 kcal/mol or more in 20 cases.

The heuristic produced the lowest energy in all but 4 cases, with differences in those cases of 0.01, 0.10, 0.70, and 1.69 kcal/mol from the best energy. In the last two cases, CFN produced the best energy. Plain MC found the best energy in only 12 cases, but gave only moderate energy errors : in just 4 cases was its best sequence more than 2 kcal/mol above the overall best energy (differences of 2.2, 2.5, 2.8, and 7.7 kcal/mol). REMC was applied to the 13 cases where the MC errors were largest ; in 4 of these it reduced the error to 0.6 kcal/mol or less. The largest MC error was reduced from 7.7 to 2.4 kcal/mol.

Doubling the REMC trajectory length reduced the two largest remaining errors to 1.1 and 1.8 kcal/mol.

3.3.3 Density of states above the GMEC

The exact CFN method can enumerate exhaustively sequence/conformation states above the GMEC, up to a given energy threshold, if the threshold is not too large. Monte Carlo and REMC explore states randomly, within a typical energy range that depends on temperature. To characterize the diversity of the sequence ensembles, we focus on the CFN and REMC methods, and we consider both the sequence entropy and the total number of states.

The mean, exponentiated sequence entropies $\langle e^{S_i} \rangle$ are reported in Table 3.6 for each test protein. The sequence entropies for the corresponding Pfam alignments (both seed and full) are also shown. The values are averaged over the designed positions in the protein chain, and can be interpreted as a mean number of amino acid classes sampled at each position. There are six classes (see Methods), one of which (Gly) is not available to the designed positions but is present in Pfam. The entropies are much smaller in the designed sets than in the Pfam sets. Retaining the top 10,000 designed sequences, CPD samples 1.3 to 1.7 amino acid classes at each position on average, compared to 3–4 in the Pfam alignments. Thus the CPD sets are much less diverse than Pfam, as observed earlier for these and other protein families ???. However, we showed earlier that if we did CPD for around ten backbone conformations, corresponding to ten representatives of a particular domain class (SH3, SH2, PDZ), then collected the sampled sequences, the overall entropy was similar to Pfam ???.

The entropy $S(E)$ is shown in Fig. 6 for the 1CKA SH3 protein as a function of the energy threshold E . Exact CFN results are compared to REMC. For this small protein, complete enumeration was feasible up to an energy threshold of $E = 2$ kcal/mol above the GMEC. REMC samples energies up to about 14 kcal/mol above the GMEC. The REMC sampling is essentially complete up to about 0.75 kcal/mol above the GMEC, at which point the REMC curve (grey) begins to depart from the exact, CFN curve (black). However, the REMC diversity at each position agrees very well with the CFN result up to about 1.5 kcal/mol above the GMEC. At $E = 2$ kcal/mol above the GMEC, the REMC entropy is still 93% of the exact value. Thus, REMC samples the full sequence diversity at each position in this range, even though it does not sample exhaustively all the combinations of mutations (let alone rotamers) at all positions.

As we consider higher energy threshold values, $E \geq 3$ kcal/mol, the number of states sampled by REMC increases exponentially and the entropy increases in a quasilinear way. Different replicas sample different energy ranges, as expected ; for example, the $kT=0.592$ and $kT=0.888$ kcal/mol replicas sample the 4–10 and 11–14 kcal/mol ranges, respectively.

3.4 Conclusions

Stochastic search methods are very common in CPD. They can be extended to very large search spaces that include backbone flexibility ???, and they do not directly depend on additive energy functions (although additivity can dramatically increase efficiency). In contrast, while exact methods like CFN have been extended beyond purely additive energy functions ?? and discrete rotamer sets ??, they are less transferable than MC and REMC. REMC is also a powerful engine to explore sequence diversity, with energy ranges of 10 kcal/mol or more above the GMEC readily accessible in this work.

Here, we tested three stochastic search methods, and compared their ability to identify low energy sequences in problems of increasing size/complexity. Direct comparison to the exact GMEC was possible routinely for problems involving up to 10 designed positions, and for 28 of 45 tests with 20 designed positions. The 10-position designs involved total rotamer numbers of 2500–3000 ; for the 20- and 30-position designs, there are about 2000 and 4000 more rotamers, respectively. The larger tests are relevant for whole protein design projects, as well as projects that redesign one or more large protein surfaces (eg, protein crystal design). For these tests, the GMEC was usually not available, and so the stochastic methods could only be evaluated indirectly. Here, the indirect quality indicators were consistency between the methods and general sequence quality compared to experiment. Indeed, agreement between the heuristic and REMC solutions was very good in general, and agreement with experimental Pfam sequences was excellent for core residues, as observed previously with the same energy function (but a heuristic exploration method) ?? . Results with a more advanced protein force field and Generalized Born solvent are expected to be even better ?? . Designed surface residues were less similar to experiment, which is at least partly because the experimental sequences are subject to additional constraints and selective pressures (such as domain-domain interactions), not included in the design.

Overall, the heuristic and REMC methods gave very good agreement with each other and with the GMEC when available. CFN, in its Toulbar2 implementation was very effective for up to 10 designed positions. Exploration speed was similar for all methods for 10-position design, and similar for the stochastic methods applied to the larger problems.

With 8-walker REMC and 0.75 billion steps per walker, the three largest departures from the overall best energy, among 67 difficult, large tests, were 4, 3, and 2.4 kcal/mol. Sequence diversity was also recapitulated accurately, compared to exact enumeration. We have recently extended the method to allow backbone moves, with the help of a hybrid move scheme to be described elsewhere. Overall, both the heuristic and REMC method appear to be effective search methods for all problem sizes.

Acknowledgements

We thank Seydou Traoré for help with the Toulbar2 program and Georgios Archontis, Isabelle André, and Sophie Barbe for helpful discussions.

Table 3.1 – Selected MC and REMC protocols

name	walker temperature(s) kT (kcal/mol)	trajectory length (steps)	move probabilities ^a rot ; mut ; mut+rot ; mut+mut ;	walker swap periodicity ^b
MC	0.2	$6 \cdot 10^9$	0; 1; 0.1; 0	-
REMCa	0.125 ; 0.25 ; 0.5 ; 1	$1.5 \cdot 10^9$	1; 0; 0.1; 0	$7.5 \cdot 10^6$
REMCb	0.25 ; 0.5 ; 1 ; 2	$1.5 \cdot 10^9$	1; 0; 0.1; 0	$7.5 \cdot 10^6$
REMCc	0.175 ; 0.263 ; 0.395 ; 0.592 ; 0.888 ; 1.333 ; 2 ; 3	$0.75 \cdot 10^9$	0; 1; 0.1; 0	$7.5 \cdot 10^6$
REMCd	0.175 ; 0.263 ; 0.395 ; 0.592 ; 0.888 ; 1.333 ; 2 ; 3	$0.75 \cdot 10^9$	1; 0; 0.1; 0	$7.5 \cdot 10^6$
REMCe	0.175 ; 0.263 ; 0.395 ; 0.592 ; 0.888 ; 1.333 ; 2 ;	$0.75 \cdot 10^9$	1; 0; 0.1; 0	10^6

^aProbabilities at each MC step to change, respectively, a rotamer ; a sidechain type ; a type at one position and a rotamer at another ; a type at two positions. ^bThe interval between attempts to exchange states between two walkers (using a Metropolis test).

Chapitre 3. Comparing stochastic search algorithms for computational protein design

Table 3.2 – Test proteins

type	PDB	length	acronym	type	PDB	length	acronym
PDZ	1G9O	91	NHERF	SH2	1A81	108	Syk kinase
PDZ	1R6J	82	syntenin	SH2	1BM2	98	Grb2
PDZ	2BYG	97	DLH2	SH2	1M61	109	Zap70
SH3	1ABO	58	Abl	SH2	1O4C	104	Src kinase
SH3	1CKA	57	c-Crk				

Table 3.3 – Designed sequence quality measures

Protein	Number of sequences tested	Identity % to wildtype	Superfamily tests				
			Match length	Superfamily E-value	Superfamily success rate	Family E-value	Family success rate
1A81	236	27	none				
1ABO	203	32	51/58	4.4e-4	100%	2.8e-3	100%
1BM2	209	27	78/98	4.2e-5	100%	2.6e-3	100%
1CKA	416	33	40/57	1.1e-5	100%	3.4e-3	100%
1G9O	338	36	79/91	7.0e-7	100%	2.5e-3	100%
1M61	405	42	97/109	7.2e-7	100%	2.6e-4	100%
1O4C	274	21	95/104	2.1e-4	100%	4.5e-3	100%
1R6J	270	34	74/82	9.8e-6	100%	4.6e-3	100%
2BYG	426	28	59/97	1.4e-5	100%	7.1e-3	100%

3.4. Conclusions

Table 3.4 – Tests with 10 designed positions

rotamers ^a	length ^b	Protein	CFN ^c	Heur. ^d	MC	REMC
2991	108(17)	1A81 3	gmec	0.001	0.1595	
		1A81 4	gmec	0.	0.0317	
		1A81 5	gmec	0.	0.0563	
2520	58(8)	1ABO 1	gmec	0.0675	0.9054	0.8041
		1ABO 4	gmec	0.	0.0128	
2957	98(10)	1BM2 1	gmec	0.	0.0950	
		1BM2 5	gmec	0.	0.1082	
2508	57(8)	1CKA 5	gmec	0.2859	3.2525	0.
2819	91(15)	1G9O 3	gmec	0.1366	0.1366	
		1G9O 5	gmec	0.	3.9599	0.
2957	109(21)	1M61 1	gmec	0.	0.0776	
		1M61 2	gmec	3.5105	4.5062	0.3215
		1M61 5	gmec	0.	0.0432	
3037	104(8)	1O4C 1	gmec	0.	0.1121	
		1O4C 2	gmec	0.	0.1046	
		1O4C 3	gmec	0.	0.1519	
		1O4C 4	gmec	0.	0.1545	
		1O4C 5	gmec	0.	0.1753	
2773	82(10)	1R6J 1	gmec	0.	2.4022	0.3986
		1R6J 2	gmec	0.	1.0398	0.3049
		1R6J 3	gmec	0.	0.0106	
		1R6J 5	gmec	0.	0.0162	
2888	97(15)	2BYG 1	5.7485	0.	0.0337	
		2BYG 3	gmec	0.	0.0833	
		2BYG 4	gmec	0.	0.2149	

^aTotal number of rotamers available to the system. Each designed position can explore 206 rotamers; the others explore about 10 rotamers each. ^bTotal protein length (number of Gly+Pro in parentheses). ^cgmec indicates the GMEC was successfully identified. ^dFor all four exploration methods and each test, we report the difference between the best energy obtained and the overall best energy (the best over all methods, which may or not be the GMEC). 10-position tests where all four methods found the GMEC are not listed.

Table 3.5 – Tests with 20 and 30 designed positions

Protein	20 positions					30 positions			
	CFN	Heur.	MC	REMC	mutations ^a	CFN	Heur.	MC	REMC
1A81 1	gmec*	0.	0.3275	0.3851	0	1.2074	0.	0.6353	
1A81 2	gmec*	0.1705	2.4355	1.0069	3	2.5520	0.	0.0578	
1A81 3	gmec	0.	0.4640	0.6186	0	43.5263	0.	2.4996	1.2025
1A81 4	gmec	0.3878	0.5748	0.6991	4	5.1300	0.	0.0305	
1A81 5	gmec	0.0068	0.5088	0.1541	4	3.2417	0.	1.9586	0.5791
1ABO 1	gmec	0.1205	1.1159	0.2153	2	44.5504	0.	0.	
1ABO 2	13.8563	0.	0.	0.	8	12.7303	0.	0.	
1ABO 3	1.2190	0.	0.	0.	9	9.3870	0.	0.2630	
1ABO 4	1.9940	0.	0.0076	0.	5	10.7691	0.	0.	
1ABO 5	3.5418	0.	0.9483	0.9483	9	4.3907	0.	0.	
1BM2 1	gmec	0.	0.0619	0.1584	0	22.5876	0.	1.7290	1.6013
1BM2 2	7.5304	0.	0.0725	0.0143	8	22.1386	0.	1.9856	1.5876
1BM2 3	gmec	0.0229	0.4762	0.2897	0	22.5410	0.	1.9990	1.1541
1BM2 4	0.1186	0.	2.5883	0.0789	2	15.2639	0.	2.2127	2.3854
1BM2 5	gmec	0.2396	0.3746	0.3746	3	15.9890	0.	2.8354	1.1937
1CKA 1	gmec*	0.	0.	0.	0	6.2700	0.	0.	
1CKA 2	gmec	0.	0.	0.	0	2.0995	0.	0.	
1CKA 3	gmec	0.	0.	0.	0	47.0217	0.	0.	
1CKA 4	4.3122	0.	0.	0.	4	44.0830	0.	0.	
1CKA 5	4.2849	0.	0.	0.	3	8.8608	0.	0.	
1G9O 1	2.0574	0.	1.2525	1.2525	5	2.0816	0.	1.5942	0.
1G9O 2	3.2106	0.	0.2177	0.1915	1	0.3270	0.	0.3126	
1G9O 3	1.9008	0.	0.4417	0.1019	1	17.7150	0.	1.5667	1.5667
1G9O 4	0.5030	0.	0.3855	0.1455	5	2.9758	0.	1.4284	1.6202
1G9O 5	0.4298	0.	0.1495	0.5114	5	0.	1.6890	7.6985	2.3857
1M61 1	gmec	0.	0.	0.	0	14.4935	0.0097	0.	0.
1M61 2	gmec	0.	0.	0.	0	5.0899	0.	1.8749	0.008
1M61 3	gmec	0.	0.	0.	0	3.5795	0.	0.0154	
1M61 4	gmec	0.	0.	0.	0	16.1511	0.	0.	
1M61 5	gmec	0.	0.2521	0.1345	0	23.0927	0.	0.	
1O4C 1	0.	0.3465	0.0690	0.0587	6	14.9064	0.	0.3435	
1O4C 2	6.4214	0.	0.1963	0.3175	4	58.1558	0.	0.0795	
1O4C 3	gmec	0.	0.3461	0.0997	0	9.9221	0.	0.1789	
1O4C 4	gmec	0.	0.3640	0.1382	0	5.7790	0.	0.0423	
1O4C 5	0.	0.	0.1131	0.2206	0	9.9221	0.	0.1789	
1R6J 1	gmec	0.	0.2604	0.2002	0	gmec*	0.	0.0246	
1R6J 2	gmec	0.	0.0071	0.0183	0	14.9800	0.	0.0957	
1R6J 3	gmec	0.	0.0537	0.0732	0	0.	0.	0.0440	
1R6J 4	gmec	0.	0.0639	0.0601	0	0.	0.	0.0957	
1R6J 5	gmec	0.	0.0735	0.0244	0	0.	0.7036	1.8823	0.0781
2BYG 1	gmec	0.	3.1878	0.0257	0	17.9752	0.	0.1592	
2BYG 2	gmec	0.	0.0524	0.0831	0	0.3832	0.	0.1502	
2BYG 3	gmec*	0.	1.3564	0.0826	0	0.1442	0.	0.1593	
2BYG 4	gmec	0.	0.1968	0.6022	0	0.	0.0958	0.0050	
2BYG 5	1.8604	0.	0.0933	0.0386	2	0.5003	0.	0.6876	

Format as in Table 3.4. gmec* indicates the more aggressive protocol. ^aBetween CFN/Heur.

Table 3.6 – Designed and Pfam sequence entropies

Protein	Top 10,000 structures	Top 10,000 sequences	Pfam seed	Pfam full
1ABO	1.36	1.58	2.79	3.01
1CKA	1.20	1.41	2.84	3.03
1R6J	1.33	1.48	3.11	3.66
1G9O	1.21	1.53	3.29	3.81
2BYG	1.57	1.63	3.31	3.67
1BM2	1.08	1.26	2.90	3.50
1O4C	1.36	1.68	2.94	3.47
1M61	1.31	1.41	2.91	3.51
1A81	1.13	1.29	2.91	3.51

The entropies are exponentiated, then averaged over all positions. The designed entropies correspond to REMC runs where all positions are designed (except Gly/Pro).

Figure captions

1. Sequence logos for the core region of two designed proteins : 1CKA (SH3) and 2BYG (PDZ). The low energy CPD sequences are compared to the sequences of the full Pfam collection of experimental sequences. Positions shown correspond to the hydrophobic core of each protein ; residue numbers are indicated (PDB numbering).
2. Histogram of Blosum40 similarity scores to Pfam sequences for the core region of two designed proteins : 1ABO (SH3) and 1BM2 (SH2). The similarity between Pfam sequences is also shown, considering either the Pfam seed alignment or the much larger full alignment.
3. Run times for different test calculations and search methods. CPU times per core are shown ; only the REMC calculations use multiple cores (OpenMP parallelization). CFN results are only given for 20 designed positions or fewer. REMC was only done for the larger calculations. Results are shown for a large, representative subset of the test calculations. A few CFN calculations were allowed to run beyond the 24h CPU limit. For clarity, the average Heur and REMC values are included as dashed lines.
4. Comparison between selected heuristic, MC, and REMC protocols for whole protein design. The best energy obtained with each protocol is shown for the nine test proteins. Zero represents the overall best energy for each protein. The mean “error” is indicated on the right for each protocol (the mean difference from the overall best energy). The protocols are a heuristic method (Heur), an MC method, two REMC protocols with 4 walkers (REMC a, b), and three REMC protocols with 8 walkers (yellow lines and dots, REMC c, d, e). Details of the protocols are in Table 3.1 ; each curve is labelled according to the protocol name.
5. Lowest energies obtained with the different exploration methods for 5-, 10-, 20-, and 30-position design. Differences with respect to the GMEC or the overall best energy are shown, excluding the smallest values (less than 0.4 kcal/mol above the best energy). The color of each point indicates the nature of the test ; its shape indicates which method gave the best energy. Two examples are highlighted by arrows : the black arrow shows the heuristic result for a 10-position test where the best energy was given by CFN ; the grey arrow shows the MC result for a 20-position test where the best energy was given by the heuristic.
6. Sequence entropy $S(E)$ and number of states $N(E)$ within a given energy range E above the GMEC for the 1CKA SH3 domain. All positions were allowed to vary except Gly/Pro. The entropy (large dots) is a single position sequence entropy, Eq. (4.4.7), averaged over all the variable positions. CFN results (black) are based on a complete

enumeration of all states within the energy range (at most E kcal/mol above the GMEC). REMC results (grey) are based on the states sampled by all 8 walkers during a single trajectory. The number of states (small dots) corresponds to all the different combinations of sequences and rotamers.

Figure 3.1 – Sequence logos for the core region of two designed proteins : 1CKA (SH3) and 2BYG (PDZ). The low energy CPD sequences are compared to the sequences of the full Pfam collection of experimental sequences. Positions shown correspond to the hydrophobic core of each protein ; residue numbers are indicated (PDB numbering).

Figure 3.2 – Histogram of Blosum40 similarity scores to Pfam sequences for the core region of two designed proteins : 1ABO (SH3) and 1BM2 (SH2). The similarity between Pfam sequences is also shown, considering either the Pfam seed alignment or the much larger full alignment.

Figure 3.3 – Run times for different test calculations and search methods. CPU times per core are shown ; only the REMC calculations use multiple cores (OpenMP parallelization). CFN results are only given for 20 designed positions or fewer. REMC was only done for the larger calculations. Results are shown for a large, representative subset of the test calculations. A few CFN calculations were allowed to run beyond the 24h CPU limit. For clarity, the average Heur and REMC values are included as dashed lines.

Figure 3.4 – Comparison between selected heuristic, MC, and REMC protocols for whole protein design. The best energy obtained with each protocol is shown for the nine test proteins. Zero represents the overall best energy for each protein. The mean “error” is indicated on the right for each protocol (the mean difference from the overall best energy). The protocols are a heuristic method (Heur), an MC method, two REMC protocols with 4 walkers (REMC a, b), and three REMC protocols with 8 walkers (yellow lines and dots, REMC c, d, e). Details of the protocols are in Table 3.1 ; each curve is labelled according to the protocol name.

Figure 3.5 – Lowest energies obtained with the different exploration methods for 5-, 10-, 20-, and 30-position design. Differences with respect to the GMEC or the overall best energy are shown, excluding the smallest values (less than 0.4 kcal/mol above the best energy). The color of each point indicates the nature of the test ; its shape indicates which method gave the best energy. Two examples are highlighted by arrows : the black arrow shows the heuristic result for a 10-position test where the best energy was given by CFN ; the grey arrow shows the MC result for a 20-position test where the best energy was given by the heuristic.

Figure 3.6 – Sequence entropy $S(E)$ and number of states $N(E)$ within a given energy range E above the GMEC for the 1CKA SH3 domain. All positions were allowed to vary except Gly/Pro. The entropy (large dots) is a single position sequence entropy, Eq. (4.4.7), averaged over all the variable positions. CFN results (black) are based on a complete enumeration of all states within the energy range (at most E kcal/mol above the GMEC). REMC results (grey) are based on the states sampled by all 8 walkers during a single trajectory. The number of states (small dots) corresponds to all the different combinations of sequences and rotamers.

Chapitre 4

PDZ

4.1 Introduction

Nous cherchons maintenant, à évaluer la performance de notre modèle CPD sur un ensemble de protéines. Les domaines PDZ (« Postsynaptic density-95 / Discs large / Zonula occludens-1 ») sont de petits domaines globulaires qui établissent des réseaux d'interactions entre protéines dans la cellule ??????. Ils forment des interactions spécifiques avec des protéines cibles, généralement en reconnaissant quelques acides aminés à l'extrémité C-terminale. En raison de leur importance biologique, les domaines PDZ et leur interaction avec les protéines cibles ont été largement étudiés et utilisés en conception *in silico*. Des ligands ont été conçus pour moduler l'activité de domaines PDZ impliqués dans diverses pathologies ????. Des domaines PDZ et leurs ligands redessinés ont été utilisés pour élucider les principes du repliement des protéines et de l'évolution ?????. Ainsi, ces domaines avec leurs ligands peptidiques fournissent des « benchmarks » pour tester les méthodes informatiques elles-mêmes ????. À partir d'une sélection de domaines PDZ, nous optimisons les énergies de référence ,paramètres essentiels dans notre modèle, E_t^r grâce à la méthode du maximum de vraisemblance. La performance du modèle est testée en générant des séquences par Proteus pour chaque protéine de la sélection. Pour cela, nous nous basons sur les résultats précédents, en particulier ceux de la section ??12), pour définir les valeurs des paramètres de l'optimisation du Monte-Carlo. Nous confrontons nos résultats à ceux de la fonction d'énergie de Rosetta , fonction, qui a connu le plus de succès?. Elle comprend un terme de répulsion de Lennard-Jones, un terme Coulomb, un terme de liaison hydrogène,un terme de solvatation Lazaridis-Karplus et des énergies de référence d'état dépliées, mais est plus empirique que la notre. Il y a un grand nombre de paramètres spécifiquement optimisés pour CPD, qui offrent des performances optimales, mais une interprétation physique moins transparente que Proteus qui lui offre la capacité de calculer les énergies libres.

La production de nos séquences calculées a été effectuée par des simulations Monte-Carlo où toutes les positions de la chaîne polypeptidique ont été autorisées à muter librement, excepté celles occupées par une glycine ou une proline qui conservent leur type d'acide aminé. Ces exceptions sur ces deux types d'acide aminé découlent de la contrainte du backbone fixe. Nous avons alors des milliers de variantes pour chaque domaine étudié. Nos tests comprennent une validation croisée où les énergies de référence optimisées sur un sous-ensemble de notre sélection sont utilisées sur un autre sous-ensembles de cette même sélection. Nous, réalisons également, une série de simulations Monte-Carlo de deux domaines PDZ où le potentiel chimique hydrophobe des types d'acides aminés est progressivement augmenté, polarisant artificiellement la composition de la protéine. Comme le biais hydrophobe augmente, les acides aminés hydrophobes envahissent progressivement la protéine de l'intérieur, formant, à partir d'un certain seuil, un noyau hydrophobe devenu plus grand que le naturel. La propension de chaque position du noyau à devenir hydrophobe à un niveau de biais plus ou moins élevé peut être considéré comme un indice d'hydrophobicité déterminé en fonction de la structure qui nous renseigne sur la propension du cœur à supporter des mutations.

4.2 Le modèle d'état déplié

4.2.1 Les énergies de référence

L'énergie utilisée est ici l'énergie de pliage de la protéine, c'est-à-dire la différence entre son énergie à l'état replié et son énergie à l'état déplié. Un mouvement élémentaire possible est une « mutation » : Nous modifions le type de chaîne latérale $t \rightarrow t'$ à une position choisie i dans la protéine pliée, en assignant un rotamère r' particulier à la nouvelle chaîne latérale. Nous considérons la même mutation dans l'état déplié. Pour une séquence S particulière, l'énergie d'état déplié est de la forme :

$$E^u = \sum_{i \in S} E^r(t_i) \quad (4.1)$$

Ici, i est la position dans la séquence et t_i le type en i , la somme se faisant sur tous les acides aminés de S .

Les grandeurs $E^r(t)$, que nous noterons également E_t^r sont appelées « énergies de référence ». Ils peuvent être considérés comme des potentiels chimiques effectifs de chaque type d'acide aminé. Le changement d'énergie de repliement d'une mutation a donc la

forme :

$$\Delta E = \Delta E^f - \Delta E^u = (E^f(\dots t'_i, r'_i \dots) - E^f(\dots t_i, r_i \dots)) - (E^r(t'_i) - E^r(t_i)) \quad (4.2)$$

avec ΔE^f et ΔE^u les changements d'énergie dans l'état replié et déplié, respectivement.

4.2.2 La vraisemblance des énergies de référence

Les énergies de référence sont des paramètres essentiels dans le modèle de simulation. Notre objectif ici est de les déterminer empiriquement afin que la simulation produise des fréquences d'acides aminés qui correspondent à un ensemble de valeurs cibles, notamment des valeurs expérimentales.

Pour cela, nous choisissons E_t^r qui maximisent la probabilité de Boltzmann d'un ensemble de séquences expérimentales. C'est à dire, nous retenons les E_t^r les plus vraisemblables étant donnée l'observation des séquences expérimentales.

Soit S une séquence particulière. Sa probabilité de Boltzmann est :

$$p(S) = \frac{1}{Z} \exp(-\beta \Delta G_S), \quad (4.3)$$

où $\Delta G_S = G_S^f - E_S^u$ est l'énergie libre de repliement de S , G_S^f est de l'énergie libre de l'état replié, $\beta = \frac{1}{kT}$ est la température inverse et Z une constante de normalisation (la fonction de partition). Nous avons alors

$$kT \ln p(S) = \sum_{i \in S} E^r(t_i) - G_S^f - kT \ln Z = \sum_{t \in aa} n_S(t) E_t^r - G_S^f - kT \ln Z, \quad (4.4)$$

où la somme à droite se fait sur l'ensemble des types d'acides aminés et $n_S(t)$ est le nombre d'acides aminés de type t dans S .

Nous considérons maintenant un ensemble \mathcal{S} de N séquences cibles ; on appelle \mathcal{L} la probabilité d'observer l'ensemble entier. \mathcal{L} est fonction des paramètres du modèle E_t^r . Comme nous voulons le maximum de \mathcal{L} sur les E_t^r , nous nous référons à \mathcal{L} comme la vraisemblance des E_t^r ?.

Nous avons :

$$kT \ln \mathcal{L} = \sum_S \sum_{i \in aa} n_S(t) E_t^r - \sum_S G_S^f - N kT \ln Z = \sum_{t \in aa} N(t) E_t^r - \sum_S G_S^f - N kT \ln Z, \quad (4.5)$$

avec $N(t)$ le nombre d'acides aminés de type t dans l'ensemble \mathcal{S} . Le facteur de normalisation Z est une somme sur l'ensemble les séquences possibles R :

$$Z = \sum_R \exp(-\beta \Delta G_R) = \sum_R \exp(-\beta \Delta G_R^f) \prod_{t \in aa} \exp(\beta n_R(t) E_t^r) \quad (4.6)$$

Pour maximiser \mathcal{L} , nous considérons la dérivé de Z selon chacune des E_t :

$$\frac{\partial Z}{\partial E_t^r} = \sum_R \beta n_R(t) \exp(-\beta \Delta G_R^f) \prod_{s \in aa} \exp(\beta n_R(s) E_s^r) \quad (4.7)$$

Nous avons alors :

$$\frac{kT}{Z} \frac{\partial Z}{\partial E_t^r} = \frac{\sum_R n_R(t) \exp(-\beta \Delta G_R)}{\sum_R \exp(-\beta \Delta G_R)} = \langle n(t) \rangle. \quad (4.8)$$

La quantité à droite est la moyenne de Boltzmann du nombre $n(t)$ des acides aminés t sur toutes les séquences possibles. C'est à dire l'espérance mathématique de $n(t)$ selon la probabilité de Boltzman. Mais comme une simulation Monte-Carlo converge vers la distribution de Boltzman, la population moyenne de t que nous obtenons dans nos simulations converge vers la moyenne de Boltzman de $n(t)$. En pratique, nous obtenons cette quantité comme la population moyenne de t dans une simulation Monte-Carlo assez longue.

Pour que $\ln \mathcal{L}$ soit maximal il faut que ses dérivées par rapport à E_t^r soient nulles.

$$\frac{1}{N} \frac{\partial}{\partial E_t^r} \ln \mathcal{L} = \frac{1}{N} \sum_S n_S(t) - \langle n(t) \rangle = \frac{N(t)}{N} - \langle n(t) \rangle \quad (4.9)$$

et donc

$$\mathcal{L} \text{ maximum} \implies \frac{N(t)}{N} = \langle n(t) \rangle, \forall t \in aa$$

Ainsi, pour maximiser \mathcal{L} , nous choisissons les E_t^r tels qu'une longue simulation donne les mêmes fréquences d'acides aminés que l'ensemble cible.

4.2.3 Recherche du maximum de vraisemblance

Nous utilisons trois méthodes pour approcher les valeurs E_t^r .

1. La première consiste à avancer dans la direction du gradient de $\ln(\mathcal{L})$ en utilisant la règle itérative suivante ?, nous l'appelons méthode linéaire :

$$E_t^r(n+1) = E_t^r(n) + \alpha \frac{\partial}{\partial E_t^r} \ln(\mathcal{L}) = E_t^r(n) + \delta E (n_t^{exp} - \langle n(t) \rangle_n) \quad (4.10)$$

avec α une constante, $n_t^{exp} = \frac{N(t)}{N}$ la population moyenne d'acide aminé de type t dans l'ensemble ciblé, $\langle \cdot \rangle_n$ indique une moyenne sur une simulation effectuée en utilisant les énergies de références courantes $E_t^r(n)$, et δE une constante empirique avec la dimension d'une énergie, correspondant à l'amplitude de mise à jour. Cette procédure de mise à jour est répétée jusqu'à convergence. Nous appelons cette méthode, la méthode de mise à jour linéaire.

2. La deuxième méthode est une variante de la première dans laquelle le δE n'est pas constant, mais ajusté au cours de la simulation de la façon suivante. On introduit une fonction proxy C comme outil de mesure rapide de l'état de l'optimisation, de la façon suivante :

$$C = \sum_{t \in aa} (n_t^{exp} - \langle n(t) \rangle_n)^2 \quad (4.11)$$

Alors, la règle (11) est utilisée trois fois avec trois valeurs différentes pour le δE ceci avec un jeu d'énergie de références identiques. Une interpolation parabolique est effectuée sur les trois valeurs de la fonction C obtenues, le minimum de la parabole est calculé et est utilisé comme δE pour le cycle suivant, au terme duquel les énergies sont mises à jour.

3. La troisième méthode, utilisée précédemment ??, utilise une règle de mise à jour logarithmique :

$$E_t^r(n+1) = E_t^r(n) + kT \ln \frac{\langle n(t) \rangle_n}{n_t^{exp}} \quad (4.12)$$

avec kT l'énergie thermique, fixée empiriquement à 0,5 kcal/mol. Nous l'appelons la méthode logarithmique. Dans les dernières itérations, certaines valeurs ont tendance à converger lentement, avec des oscillations. Par conséquent, une règle modifiée où une énergie au cycle n et l'énergie au cycle n-1 sont moyenées avec un poids respectif de 2/3 et 1/3.

Chaque itération, pour le modèle NEA (voir plus bas), a été effectuée avec 500 millions de pas par réplique de REMC, et 100 millions de pas pour le modèle FDB.

4.3 Méthodes de calcul

4.3.1 Fonction énergétique efficace pour l'état replié

La matrice énergétique a été calculée avec la fonction d'énergie suivante pour État plié :

$$E = E_{bonds} + E_{angles} + E_{dihe} + E_{impr} + E_{vdw} + E_{Coul} + E_{solv} \quad (4.13)$$

Les six premiers termes de l'équation Eq. (4.13) représentent l'énergie interne de la protéine. Ils sont tirés de la fonction d'énergie empirique Amber ff99SB ?, légèrement modifiée pour le CPD.

Le dernier terme à droite de l'équation (13), E_{solv} , représente la contribution du solvant. Nous avons utilisé un modèle de solvant implicite « Generalized Born + Surface Area » ou GBSA (44) :

$$E_{solv} = E_{GB} + E_{surf} = \frac{1}{2} \left(\frac{1}{\epsilon_w} - \frac{1}{\epsilon_p} \right) \sum_{i,j} q_i q_j (r_{ij}^2 + b_i b_j \exp[-\frac{r_{ij}^2}{4b_i b_j}])^{-\frac{1}{2}} + \sum_i \sigma_i A_i$$

Ici, ϵ_w et ϵ_p sont les constantes diélectriques du solvant et de la protéine ; r_{ij} est la distance entre les atomes i , j et b_i est le « rayon de solvatation » de l'atome i ?. A_i est la surface exposée accessible au solvant de l'atome i . σ_i est un paramètre qui représente la préférence de chaque atome à être exposé ou caché du solvant. Les atomes du soluté sont divisés en quatre groupes avec pour chacun une valeur σ_i spécifique en $\text{cal/mol}/\text{\AA}^2$: non polaire -5, aromatique -40, polaire -80, ionique -100.

On attribue aux atomes d'hydrogène un coefficient de surface de 0. Les surfaces sont calculées par l'algorithme de Lee et Richards ?, qui est implémenté dans le programme XPLOR ?, en utilisant un rayon de « probe radius » de 1,5 AA. Les simulations MC utilisent une constante diélectrique $\epsilon_p = 4$ où 8 (voir la partie Résultats).

Dans le terme énergétique GB, le rayon de solvatation atomique b_i approxime la distance de i à la surface de la protéine. C'est une fonction des coordonnées de tous les atomes de protéines. La forme b_i correspond à une variante GB que nous appelons GB/HCT, d'après ses auteurs ?, avec les paramètres du modèle optimisés pour une utilisation avec le champ de force Amber ?. Comme b_i dépend des coordonnées de tous les atomes du soluté ?, une approximation supplémentaire est nécessaire pour rendre le terme énergétique GB additif par paire et pour rendre la matrice d'énergie définissable. Pour cela, nous utilisons deux approximations, la méthode NEA pour « Native Environment Approximation » et la méthode FDB « Fluctuating Dielectric Boundary » ?.

Dans le modèle NEA, le rayon de solvatation b_i de chaque groupe (backbone, chaîne latérale ou ligand) est calculé à l'avance , le reste du système étant fixé à sa séquence et sa conformation native ??, voir le chapitre CPD.

Dans le modèle FDB, est exploité le fait que dans le GB , l'environnement diélectrique d'une paire de résidus est complètement caractérisé par un petit ensemble de rayons de solvatation atomique et ces rayons sont eux-mêmes sommes de paires sur les atomes de la protéine. Cette méthode est composée de deux étapes. La première est le calcul de rayon de solvatation moyen B_I pour chaque résidu I et la seconde est l'expression de l'interaction énergétique GB de chaque paire de résidus I , J , comme une série de puissance des B_I et B_J , voir le chapitre CPD.

La contribution de l'énergie de surface E_{surf} n'est pas non plus additive par paire, car dans la structure de la protéine, la surface enfouie par une chaîne latérale peut également être enfouie par une autre chaîne. Alors, nous avons utilisé la méthode de Street et al ?. Dans laquelle, la surface enfouie d'une chaîne latérale est calculée en additionnant la chaîne latérale voisine et les groupes backbones. Pour chaque groupe voisin, la zone de contact avec la chaîne latérale en question est calculée indépendamment des autres groupes. Les zones de contact sont additionnées. Pour éviter de sur évaluer la surface enfouie, un facteur est appliqué aux zones de contact des chaînes latérales impliquées. Des études précédentes ont montré qu'un facteur de 0,65 fonctionne bien ??.

La fonction d'énergie empirique Amber ff99SB (42), légèrement modifiée pour le CPD, en remplaçant les charges du backbone par un ensemble unifié, obtenu en faisant la moyenne sur l'ensemble des types d'acides aminés et ajuster légèrement pour rendre la partie backbone de chaque acide aminé neutre ?.

4.3.2 Les énergies de référence de l'état déplié

Dans le modèle CPD, l'énergie de l'état déplié dépend de la composition de la séquence par l'ensemble des énergies de référence E_t^r (équation 4.1). Ici, les énergies de référence ont été attribuées en fonction des types d'acides aminés t, mais aussi de la position de chaque acide aminé dans la structure repliée à travers son caractère enfoui ou exposé au solvant. Ainsi, pour un type donné (Ala, par exemple), il y a deux valeurs distinctes de E_t^r , une enfouie et une exposée. Cette approche se justifie par trois éléments. Tout d'abord, nous supposons que la structure résiduelle est présente dans l' état déplié, de sorte que les acides aminés conservent en partie leur caractère enfui/exposé. Deuxièmement, nous supposons que le modèle d'état déplié compense de manière systématique des erreurs dans la fonction d'énergie de l'état plié, de sorte que la structure pliée contribue indirectement

aux énergies de référence. Troisièmement, cette stratégie rend le modèle moins sensible aux variations de la longueur des boucles de surface et au ratio de résidus de surface sur enterrés, qui peut varier considérablement selon les homologues (voir plus bas).

Par conséquent, le modèle devrait être transférable à l'intérieur d'une famille de protéines. Distinguer les positions enfouies / exposées double le nombre de paramètres E_t^r à ajuster. Inversement, pour réduire le nombre de paramètres, nous groupons les acides aminés en classes homologues (voir table AAGroups). Dans chaque classe c , et pour chaque type de position (enfoui ou exposé), les énergies de référence ont la forme

$$E_t^r = E_c^r + \delta E_t^r$$

avec E_c^r est un paramètre ajustable,tandis que δE_t^r est une constante, calculée comme la différence d'énergie de mécanique moléculaire entre les types d'acides aminés de classe c, supposé en conformation dépliée où chaque acide aminé interagit uniquement avec lui-même et avec le solvant.

Groupe	acides aminés	propriétés
1	Ala,Cys,Thr	petit
2	Ser	
3	Glu,Asp	chargé négativement
4	Gln,Asn	polaire
5	Ile,Leu,Val	apolaire
6	Met	non polaire
7	Hip,Hid,Hie	chargé positivement
8	Arg	
9	Lys	
10	Phe,Trp	aromatique
11	Tyr	
12	Gly,Pro	non mutable

Table 4.1 – Les groupes d'acides aminés utilisés pour l'optimisation des énergies de référence.

Plus précisément, nous effectuons des simulations MC d'un peptide étendu (le peptide Syndecan1) et calculons les énergies moyennes pour chaque type d'acide aminé à chaque position peptidique (à l'exclusion des positions terminales). Nous prenons les différences entre les types d'acides aminés et les moyennons sur les positions peptidiques. Pendant, la maximisation de la vraisemblance, E_c est optimisé tandis que δE_t est fixe. Pour opti-

miser les valeurs E_t^r , nous utilisons une trois méthodes (4.2.3) avec des fréquences cibles correspondent aux fréquences expérimentales soient des classes d'acides aminés, soient des types d'acides aminés. Le choix se faisant par un début d'optimisation sur les classes, puis lorsque la convergence est correctement établie, c'est à dire lorsque la fonction proxy C calculée sur ces classes fournit des valeurs stable et faibles,nous relâchons cette contrainte pour optimiser sur l'ensemble des types d'acide aminés mutables.Typiquement, vingt cycles d'optimisation sont effectués sur les classes, puis encore vingt cycles sur les types.

4.4 Outils d'analyse de séquences

4.4.1 Superfamily/SCOP

Superfamily [?] est un ensemble composé :

- D'une base de données de modèles de Markov cachés, où chaque modèle représente une structure 3D d'un domaine de la classification SCOP.
- D'une série de scripts qui annotent à partir des informations de la base,les séquences données en entrée. Ici, nous utilisons uniquement l'association au modèle 3D le plus vraisemblable.

Nous travaillons avec la base de données à la version 1.75, et en conjonction, nous utilisons SAM (version 3.5) [?] et HMMER (version 3.0) [?] recommandés par l'équipe de Superfamily. Le paramétrage utilisé est celui par défaut. La base SCOP contient 15 438 modèles.

4.4.2 Taux d'identité de séquences

Soient S et N deux séquences d'acides aminés de même longueur l .

Le Taux d'identité $Id(S,N)$ de S par rapport N est égal au pourcentage de position où l'acide aminé est identique dans S et N . C'est-à-dire :

$$Id(S,N) = \frac{1}{l} \sum_{1 < i < l} \mathbb{1}(s_i, n_i) \times 100 \quad (4.14)$$

avec s_i et n_i l'acide animé en i de S et de N respectivement, et $\mathbb{1}(x,y)$ la fonction qui vaut 1 lorsque $x = y$ et 0 sinon.

4.4.3 Taux d'identité par position

Le taux d'identité d'un alignement A_S à la position i par rapport à une séquence N de même longueur se définit comme :

$$Id(A_S, i) = \frac{1}{m} \sum_{1 \leq j \leq m} \mathbb{1}(s_i^j, n_i) \times 100 \quad (4.15)$$

avec m le nombre de séquences de A_S .

4.4.4 Alignements Pfam

Ce taux d'identité donne une mesure de la ressemblance entre un alignement et une séquence. Cela nous permet de comparer nos séquences calculées à la séquence native. Mais cela n'est pas notre seul objectif. Et nous voulons les évaluer par rapport à l'ensemble des séquences du domaine protéique de la native. La base de données Pfam (Protein families database) [?] regroupe les domaines protéiques connus en famille. Chaque famille étant représentée par des alignements multiples de séquences et des profiles de modèles de Markov cachés [?]. Dans la suite, nous utilisons l'alignement dit « RP55 » du domaine PDZ, qui se base sur un petit alignement de membres représentatifs de la famille, l'alignement « seed » qui contient 45 séquences, et qui est augmenté grâce aux modèles de Markov cachés construits à partir de « seed », jusqu'à contenir 12 255 séquences protéiques naturelles.

4.4.5 Score BLOSUM

Pour tenir compte des ressemblances et des différences entre les acides aminés lors d'une substitution, nous avons besoin d'une matrice de coût. Nous utilisons les matrices BLOSUM40 et BLOSUM62 (BLOcks SUbstitution Matrix) [?] qui sont construites à partir de blocs d'alignement très conservés (plus de 40% et 62% d'identités respectivement). Les fréquences des mutations y sont calculées. Le score BLOSUM d'une substitution est alors le logarithme de la fréquence de la mutation correspondante. À cela est ajouté un score de pénalités pour l'insertion d'un gap (c'est-à-dire un saut dans l'alignement).

On définit alors simplement un score de similarité de deux séquences de même longueur comme la somme des scores BLOSUM62 sur toutes les positions. De même le score de similarité d'un alignement par rapport à une séquence sera défini comme la moyenne des scores de similarité sur ensemble des séquences de l'alignement. Et enfin un score de similarité de deux ensembles de séquences alignés entre eux comme la moyenne des scores de similarité du premier ensemble par rapport aux séquences du second.

4.4.6 Similarité d'un ensemble à un alignement Pfam

Afin de calculer un score de similarité d'un ensemble de nos séquences par rapport à une famille Pfam, il faut commencer par aligner nos séquences avec l'alignement de la famille. Pour cela nous utilisons le programme d'alignement BLAST [?]. Il implémente une heuristique qui recherche puis étend les meilleurs alignements locaux. Nous procédons comme suit :

1. La commande `blastp` est utilisée avec comme base de données (paramètre `-db`) l'alignement Pfam et comme séquence en entrée (paramètre `-query`) la séquence native.
2. Dans la sortie blast, la séquence qui produit l'alignement le plus significatif avec la native est collectée, notons-la S_0 .
3. L'alignement blast est alors utilisé pour positionner la native par rapport à S_0 et les gaps nécessaires pour aligner la native à S_0 sont ajoutés.
4. Le positionnement et les gaps sont alors appliqués tels quels à la liste de nos séquences.

4.4.7 Entropie par position

Pour comparer la diversité des séquences produites avec la diversité des séquences naturelles, nous utilisons l'entropie par position ?, à partir de la formule :

$$S_i = - \sum_{j=1}^6 f_j(i) \ln f_j(i) \quad (4.16)$$

avec $f_j(i)$ la fréquence du type de résidu j à la position i , au lieu de distinguer les 20 types d'acide aminé, nous utilisons six classes de résidus, correspondant à aux groupes suivants : {LVIMC}, {FYW}, {G}, {ASTP}, {EDNQ} et {KRH}. Cette classification est a été obtenu par une analyste de cluster sur la matrice BLOSUM 62 et une analyse des énergies de contact entre résidus dans les protéines ?. Pour obtenir une mesure du nombre de types d'acide aminé apparaissant à une position, on utilise l'exponentielle de l'entropie des résidus $\exp(S)$ (qui varie de 1 à 6), moyenner sur l'ensemble des résidus de la chaîne protéique. Ce qui correspond à un nombre moyen de classes de séquence échantillonnées par position. Par exemple, une valeur de 2 à une position particulière indique que les acides aminés de deux des six classes sont présents à cette position au sein de l'ensemble des séquences analysées. Une valeur moyenne globale de deux indique qu'en moyenne, deux classes d'acides aminés sont présentes à n'importe quelle position dans les séquences analysées.

4.5 Séquences expérimentales et modèles structurels

4.5.1 L'ensemble des protéines PDZ

Nous sélectionnons huit protéines de la famille PDZ dont les structures cristallographiques sont connues. Aux les trois présentes dans l'ensemble étudié au chapitre précédent : NHREF,Syntenin et DLG2, sont ajoutés les protéines INAD, GRIP, PSD95 et Cask , Tiam1.Leur séquence est présenté à la figure ???. Cela constitue un ensemble dont le nombre de positions actives, c'est-à-dire les positions qui vont être mutées , est du même ordre pour chaque séquence d'acide aminé des protéines (voir le tableau 4.2).

Table 4.2 – La sélection de domaines protéiques PDZ

nom	Code PDB	résidus	nombre de positions actives
NHREF	1G9O	9-99	76
INAD	1IHJ	13-105	82
GRIP	1N7E	668-761	79
Syntenin	1R6J	193-273	72
DLG2	2BYG	186-282	82
PSD95	3K82	305-402	80
Cask	1KWA	487-568	74
Tiam1	4GVD	838-930	84

4.5.2 Alignements Blast croisés

Pour caractériser les homologies dans cet ensemble, une série de requêtes BLAST est effectuée sur chaque paire de séquences en utilisant le programme `blastp` avec les options comme indiqué en 4.4.6. Il apparaît que Syntenin et Tiam1 sont atypiques dans l'ensemble avec, aucun homologue avec une E-value inférieure à 10^{-7} et plusieurs E-value supérieur à 10. PSD95 est la protéine la plus consensuelle, ayant d'une part une homologie avec toutes les autres à au plus $6 \cdot 10^{-4}$, et d'autre part ayant 4 homologues à moins de $2 \cdot 10^{-10}$, pour un pourcentage d'identité compris entre 30 et 46.Globalement, il n'y a que peu d'homologies, la plus forte n'étant que de $3 \cdot 10^{-15}$ entre PSD95 et DLG2 pour un pourcentage d'identité de 37. Les détails sont dans le tableau 4.3.

4.5.3 Sélection des homologues

Pour définir les fréquences d'acides aminés cibles pour maximiser nos vraisemblances, nous sélectionnons un ensemble de séquences homologues pour les 6 premières protéines

Table 4.3 – E-value et pourcentage d'identité des alignements Blast native versus native pour nos séquences PDZ.

Protein	NHREF	INAD	GRIP	Syntenin	DLG2	PSD95	CASK	TIAM1
NHREF	$2 \cdot 10^{-66}$ (100)	$5 \cdot 10^{-10}$ (40)	$2 \cdot 10^{-3}$ (25)	$3 \cdot 10^{-7}$ (25)	$2 \cdot 10^{-11}$ (35)	$1 \cdot 10^{-12}$ (30)	$5 \cdot 10^{-5}$ (25)	$9 \cdot 10^{-7}$ (35)
INAD	$5 \cdot 10^{-10}$ (40)	$3 \cdot 10^{-68}$ (100)	$2 \cdot 10^{-7}$ (27)	[18]	$2 \cdot 10^{-8}$ (27)	$9 \cdot 10^{-14}$ (46)	$4 \cdot 10^{-6}$ (35)	[16]
GRIP	$2 \cdot 10^{-3}$ (25)	$2 \cdot 10^{-7}$ (27)	$3 \cdot 10^{-67}$ (100)	[21]	$3 \cdot 10^{-14}$ (36)	$2 \cdot 10^{-10}$ (37)	$9 \cdot 10^{-12}$ (30)	$5 \cdot 10^{-5}$ (35)
Syntenin	$3 \cdot 10^{-7}$ (25)	[18]	[21]	$1 \cdot 10^{-59}$ (100)	[17]	$1 \cdot 10^{-6}$ (32)	$7 \cdot 10^{-3}$ (32)	[18]
DLG2	$2 \cdot 10^{-11}$ (35)	$2 \cdot 10^{-8}$ (27)	$3 \cdot 10^{-14}$ (37)	[17]	$7 \cdot 10^{-71}$ (100)	$3 \cdot 10^{-15}$ (37)	$2 \cdot 10^{-7}$ (28)	$5 \cdot 10^{-5}$ (41)
PSD95	$1 \cdot 10^{-12}$ (30)	$9 \cdot 10^{-14}$ (46)	$2 \cdot 10^{-10}$ (36)	$1 \cdot 10^{-6}$ (32)	$3 \cdot 10^{-15}$ (37)	$4 \cdot 10^{-70}$ (100)	$1 \cdot 10^{-7}$ (27)	$6 \cdot 10^{-4}$ (33)
Cask	$5 \cdot 10^{-5}$ (25)	$4 \cdot 10^{-6}$ (35)	$9 \cdot 10^{-12}$ (30)	$7 \cdot 10^{-3}$ (32)	$2 \cdot 10^{-7}$ (28)	$1 \cdot 10^{-7}$ (27)	$7 \cdot 10^{-61}$ (100)	$5 \cdot 10^{-4}$ (33)
Tiam1	$9 \cdot 10^{-7}$ (35)	[16]	$5 \cdot 10^{-5}$ (35)	[18]	$5 \cdot 10^{-5}$ (41)	$6 \cdot 10^{-4}$ (33)	$5 \cdot 10^{-4}$ (33)	$1 \cdot 10^{-68}$ (100)

S'il n'y a pas de touche avec une E-value inférieure à 10, [.] donne le pourcentage d'identité du couple dans l'alignement des 6 séquences sauvages.

de notre sélection (nous excluons Cask et Tiam1 pour le calcul des énergies de références). Pour cela, nous effectuons des recherches BLAST avec comme requête la séquence extraite du fichier PDB sur la base de données « siwwprot + trEmBL » d'Uniprot avec la matrice BLOSUM62 sans l'option « filtre » et avec l'option « Gapped ». Nous obtenons un premier ensemble pour chaque cas en nous limitant aux homologues de bonne qualité au regard de E-value et du pourcentage d'identité, tout en conservant en même temps une certaine diversité. Cela oblige pour certaines protéines à accepter des E-values plus haute que 10^{-40} , notamment INAD et NHREF , respectivement 10^{-32} et 10^{-10} ,pour avoir un nombre d'homologues suffisant. Ensuite, les redondances les plus flagrantes sont enlevées manuellement. Finalement,les ensembles se composent de 42 à 126 homologues,avec des pourcentages d'identité supérieurs à 66% excepté pour INAD où il a fallu descendre jusqu'à 38% d'identité. Voir le tableau pour les détails 4.4. Les alignements des séquences homologues retenues pour un groupe constitué des 6 premières protéines sont représentés aux figures ??,??,??,??,?? et ??.

Table 4.4 – Sélection des homologues.

protéines	% identité
NHREF	62 $1 \cdot 10^{-32}$ 67-95
INAD	42 $1 \cdot 10^{-10}$ 38-95
GRIP	48 $1 \cdot 10^{-45}$ 84-95
Syntenin	85 $1 \cdot 10^{-43}$ 85-95
DLG2	43 $1 \cdot 10^{-41}$ 78-95
PSD95	50 $1 \cdot 10^{-46}$ 81-95
Cask	126 $7 \cdot 10^{-28}$ 60-85
Tiam1	50 $2 \cdot 10^{-23}$ 60-85

4.5.4 Alignements des protéines expérimentales et leurs homologues

Afin d'obtenir une caractérisation structurale de notre sélection de protéines PDZ. Nous réalisons en alignement de nos huit séquences natives, présenté à la figure ???. Cet alignement nous sert de base pour la définition d'un alignement structural de nos séquences. Nous pouvons alors définir un cœur hydrophobe de nos protéines « PDZ », il est représenté en ???. Les 14 positions utilisées pour définir le cœur hydrophobe sont bien conservées dans l'alignement « seed » de Pfam, mais pas totalement. L'Arg, Lys et Gln apparaissent à certaines positions, puisque dans de petites protéines comme des domaines PDZ, la longue partie hydrophobe de ces chaînes latérales peut être enfouie dans le noyau tout en permettant à la pointe polaire de la chaîne d'être exposé au solvant. Quelques résidus Asp et Glu apparaissent aussi, dans les endroits où l'alignement des séquences peut ne pas très bien refléter la superposition 3D les chaînes latérales.

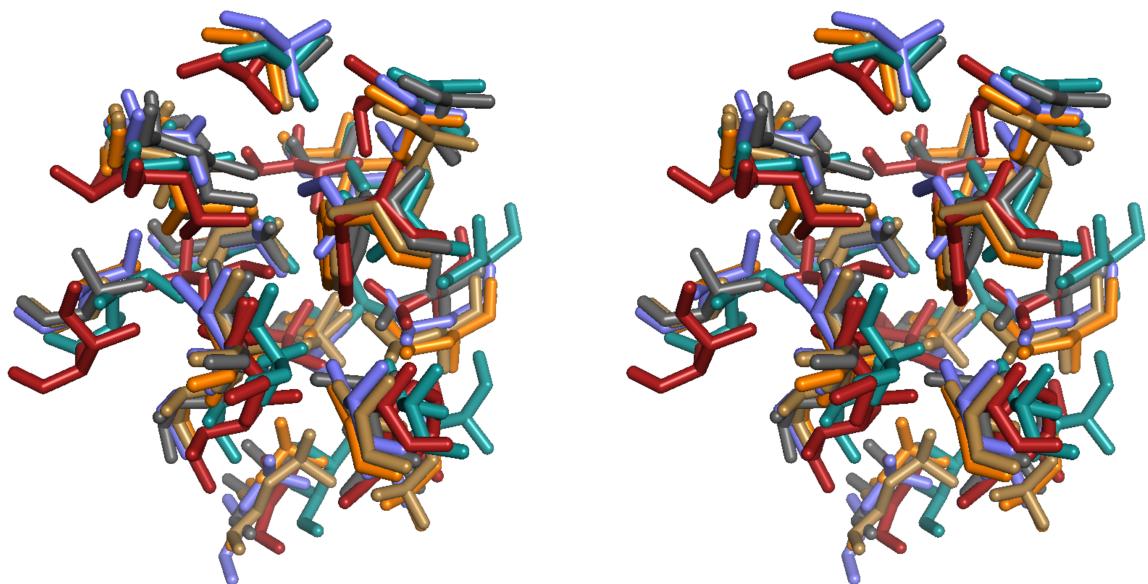
Protein	NHREF	INAD	GRIP	Syntenin	DLG2	PSD95	Cask	Tiam1
NHREF	326	64	15	15	59	112	49	1
INAD	64	221	56	-9	88	107	25	9
GRIP	15	56	378	24	65	87	90	39
Syntenin	15	-10	24	311	-26	22	42	-18
DLG2	59	88	65	-26	325	110	24	22
PSD95	112	107	87	22	110	325	66	21
Cask	49	25	90	42	23	66	308	37
Tiam1	1	10	39	-18	22	21	37	371

Table 4.5 – Similarité des séquences expérimentales homologues, pour les 8 protéines PDZ.

4.5.5 Similarité des homologues

Comme nous avons caractérisé les homologies des séquences natives, nous calculons également la similarité des séquences expérimentales homologues pour chaque couple alignées comme en ???. Et apparaît les situations où les similarités sont négatives, c'est le cas chez les homologues à Cask et Syntennin déjà repérés comme éloignés en termes de séquences, avec une valeur de -26 entre DLG2 et Syntennin, -18 entre Tiam1 et Syntennin. Les plus hautes valeurs, excepté les valeurs d'autosimilarité, sont à 112, entre PSD95 et NHREF, et 110 pour la similarité entre les protéines PSD95 et DLG2. La similarité moyenne entre deux protéines différentes de notre sélection se situe aux environs de 50.

1G90 RMLPRLCCLEK.GPNGYGFHLHGEKGKL.....GQYIRLVEPGSPAEKAG.LLAGDRIVEVNGENVEKETHQQVVSRIRALNAVRLLVVDPETDEQL
 1IHJ GELIHMVLDKTGKSFVICIVRGEVKDSPNTKTTGIFIKGIVPDSPAHLCGRLKVGDRILSLNGKDVRNSTEQAVIDLIKEADFKIELEIQT
 1N7E GAIYYTVELKR.YGGPLGITISGTEEP.....FDPIISSLTKGGLAERTGAIHGDRILA
 1R6J GAMDPRITMHKDSTGHVGIFIFKN.....GKITSIVKDSSAARNG.LLTEHNICEINGQNIVGLKDSQIA
 2BYG FQSMTVVEIKLFK.GPKGLGFSIAGGVGNQH.IPGDNSIYVTKIIDGGA
 3K82 EDIPREPRRIVIHR.GSTGLGFNIVGEGE.....GIFISFILAGGPADLSGELRKGDQILSVNGVDLNASHEQAAIALKNAGQT
 CASK RSRLVQFQKNTDEPMGIFTLKMNELN.....HCIVARIMHGGMIHROGTLHVGD
 .TIAM1 GAMGKVTHSIHIKEKSDTAADTYGFSLSSVED.....GIRRLYVNSVKETGLASKKG.LKAGDEILEINNRAADALNSSMLKDFLSQP..SLGLLVRYPEL



	Y	F	L	I	A	L	L	V	V	V	I	V	L	V
NHREF	24	26	28	39	48	53	59	62	67	75	79	86	88	90
INAD	F	I	I	I	A	L	I	L	V	V	I	I	L	I
GRIP	28	30	32	50	59	65	71	74	79	87	91	98	100	102
Syntenin	L	I	I	I	A	I	I	I	L	A	L	V	L	I
DLG2	682	684	686	698	707	713	719	722	727	735	739	746	748	750
PSD95	V	F	F	I	A	L	I	I	V	I	L	V	I	I
Cask	209	211	213	218	227	232	238	241	246	254	258	265	267	269
Tiam1	L	F	I	V	A	L	L	V	L	A	L	V	L	V
	203	205	207	224	233	239	245	248	253	261	265	272	274	276
	L	F	I	I	A	L	I	V	L	A	L	V	I	A
	323	325	327	338	347	353	359	362	367	375	379	386	388	390
	M	I	L	V	I	L	I	I	V	L	L	I	F	I
	501	503	505	515	524	530	536	539	544	552	556	563	565	567
	Y	F	L	V	A	L	I	I	A	L	L	L	L	V
	858	860	862	875	884	889	895	898	903	911	915	920	922	924

Figure 4.1 – le cœur PDZ sélectionné

4.5.6 Les fréquences d'acides aminés

Pour chaque ensemble d'homologues, notons-le H , nous calculons la moyenne sur toutes les séquences et toutes les positions pour obtenir les fréquences globales d'acides aminés. Les fréquences sont déterminées séparément pour les positions enfouies et exposées. Notons-les $f_t^b(H), f_t^e(H)$, où l'indice t représente un type d'acide aminé et les exposants e et b signifient respectivement les ensembles de positions enfouies et exposées. Enfin les ensembles de fréquences moyennes des huit protéines sont eux-mêmes moyennés sur deux groupes de protéines, d'une part le sous-ensemble $S_1 = NHREF, INAD, GRIP, Syntenin, DLG2, PSD95$ et d'autre part le groupe $S_2 = Cask, Tiam1$, ce qui donne deux jeux de deux ensembles cibles distincts de fréquences d'acides aminés f_t^b et f_t^e pour chaque type, et de même pour chaque classe de type. Cette partition en deux sous-ensembles de protéines va nous permettre d'estimer la transférabilité des énergies de références obtenues à partir d'un sous-ensemble de protéines sur un autre sous-ensemble de protéines.

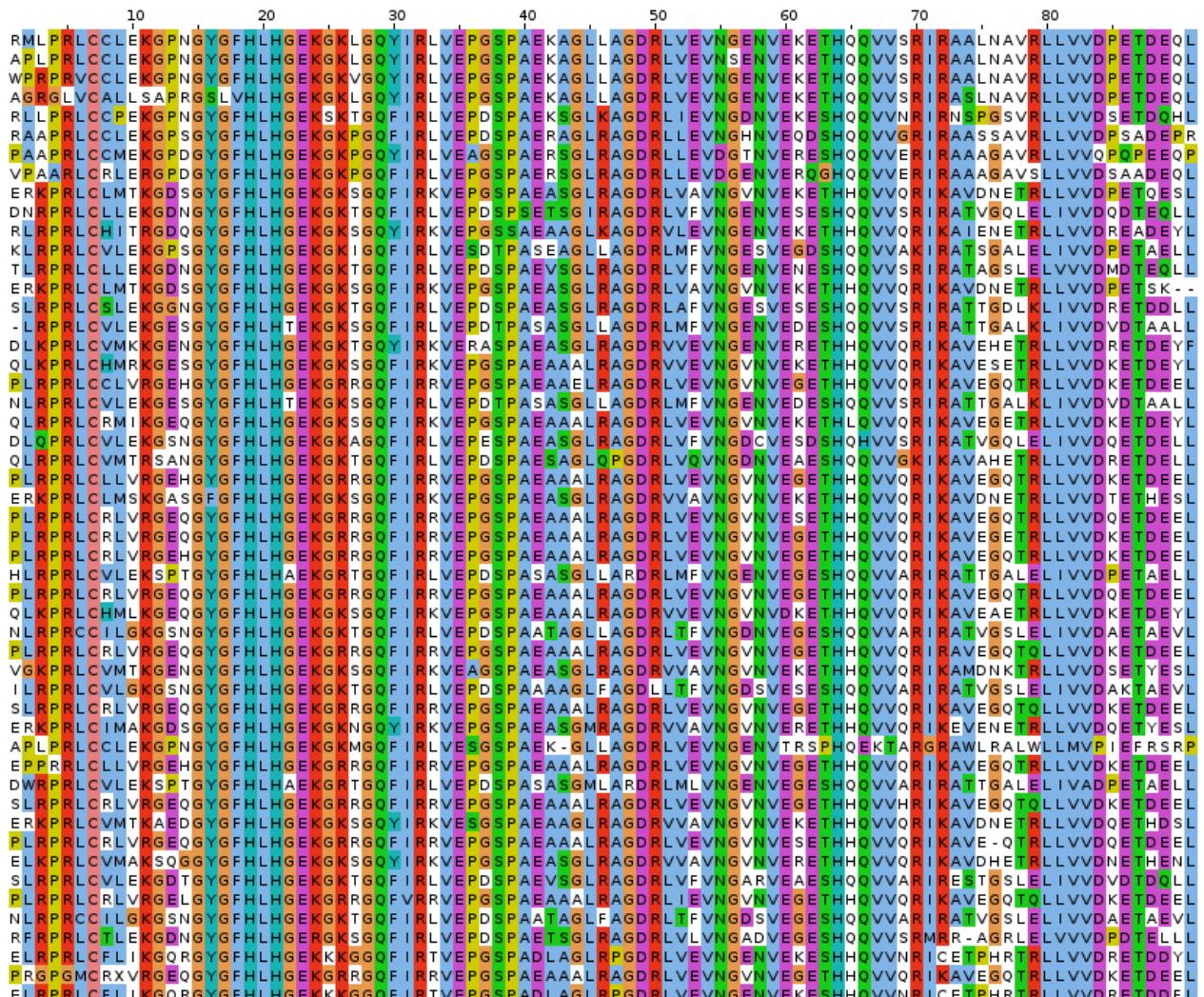


Figure 4.2 – L’alignement de notre sélection de séquences homologues à la protéine NHREF (code PDB :1G9O)

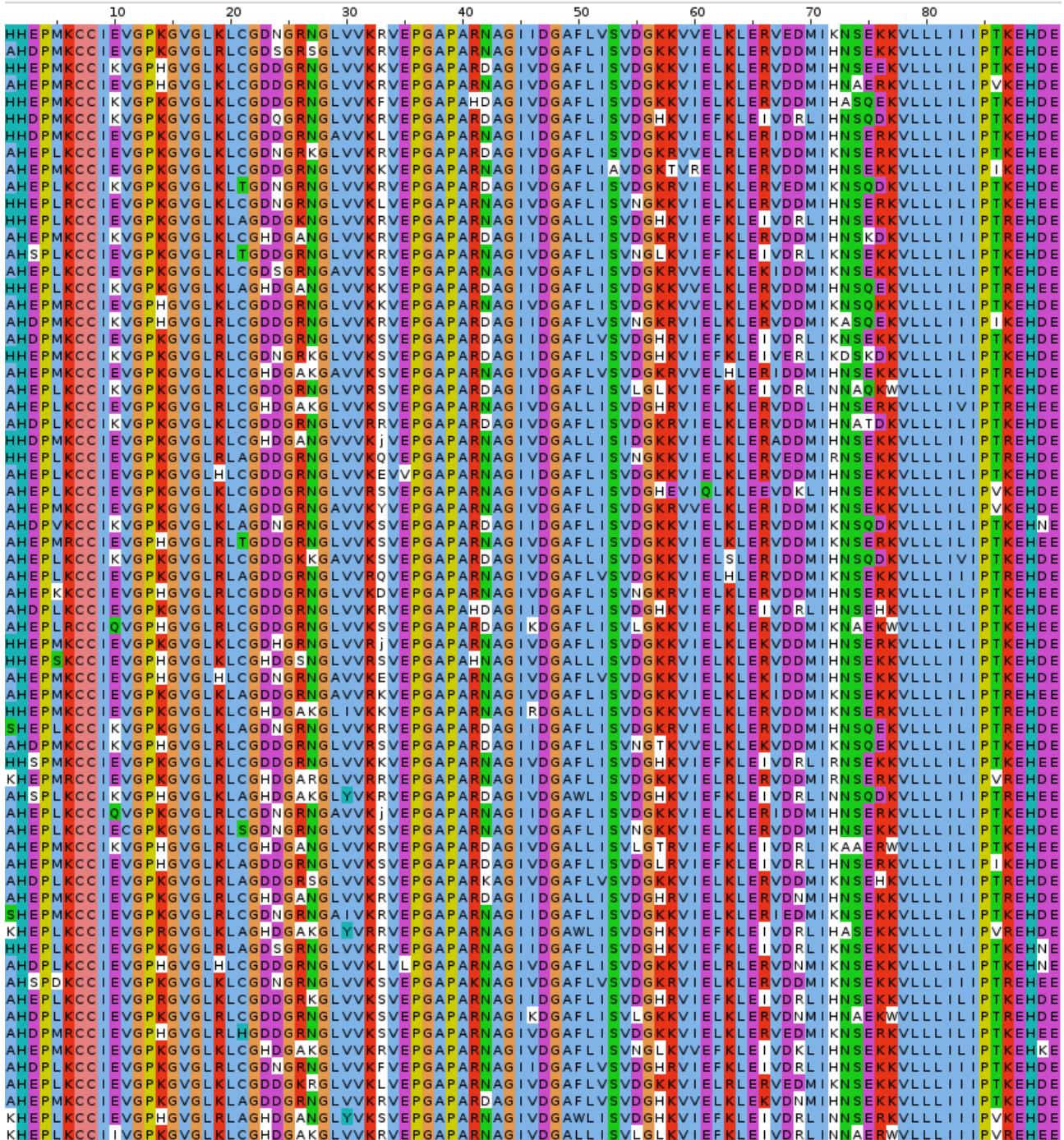


Figure 4.3 – Une sélection de séquences protéins, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine NHREF (code PDB :1G9O), modèle NEA

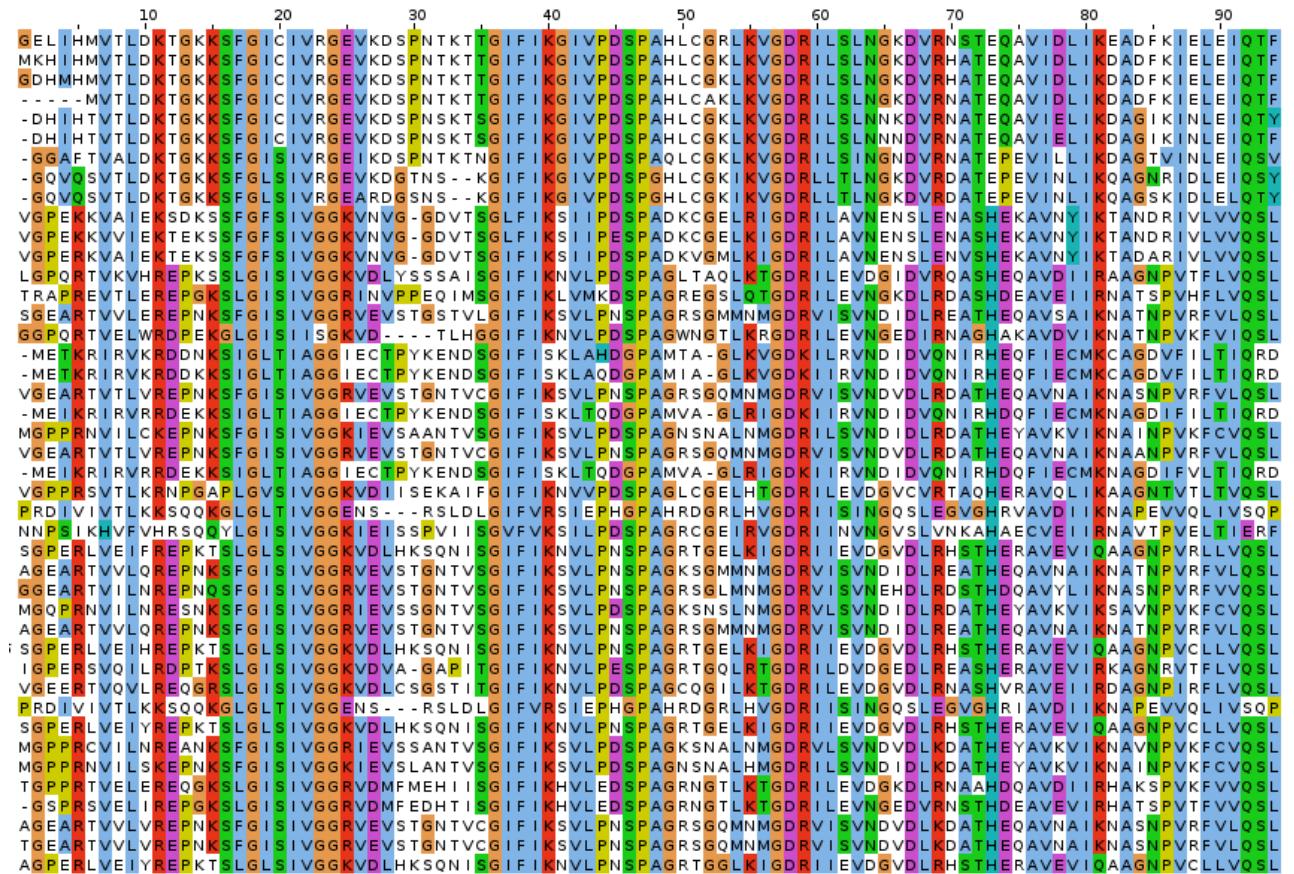


Figure 4.4 – L’alignement de notre sélection de séquences homologues à la protéine INAD (code PDB :1IHJ)

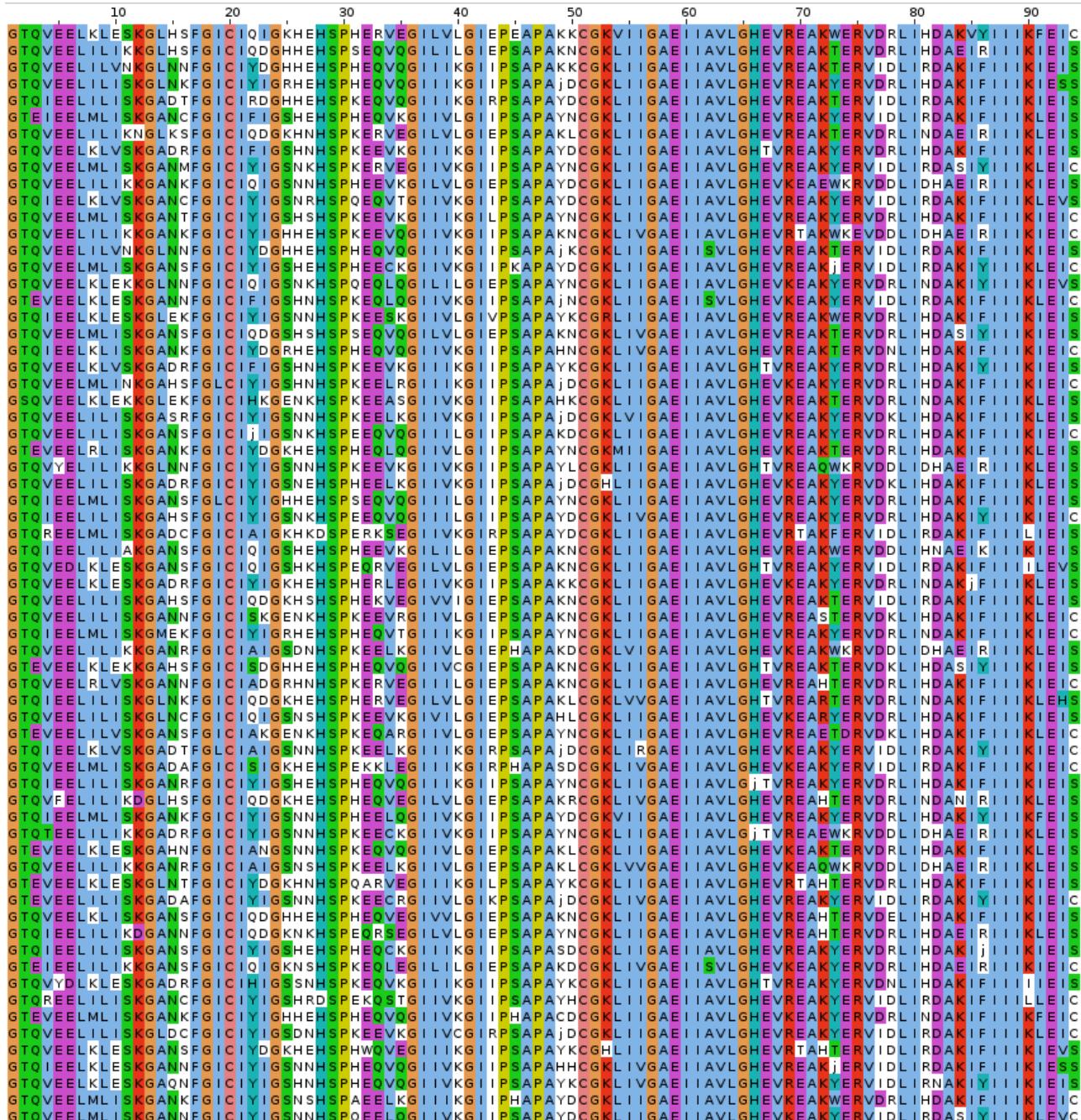


Figure 4.5 – Une sélection de séquences protéins, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine INAD (code PDB :1IHJ), modèle NEA

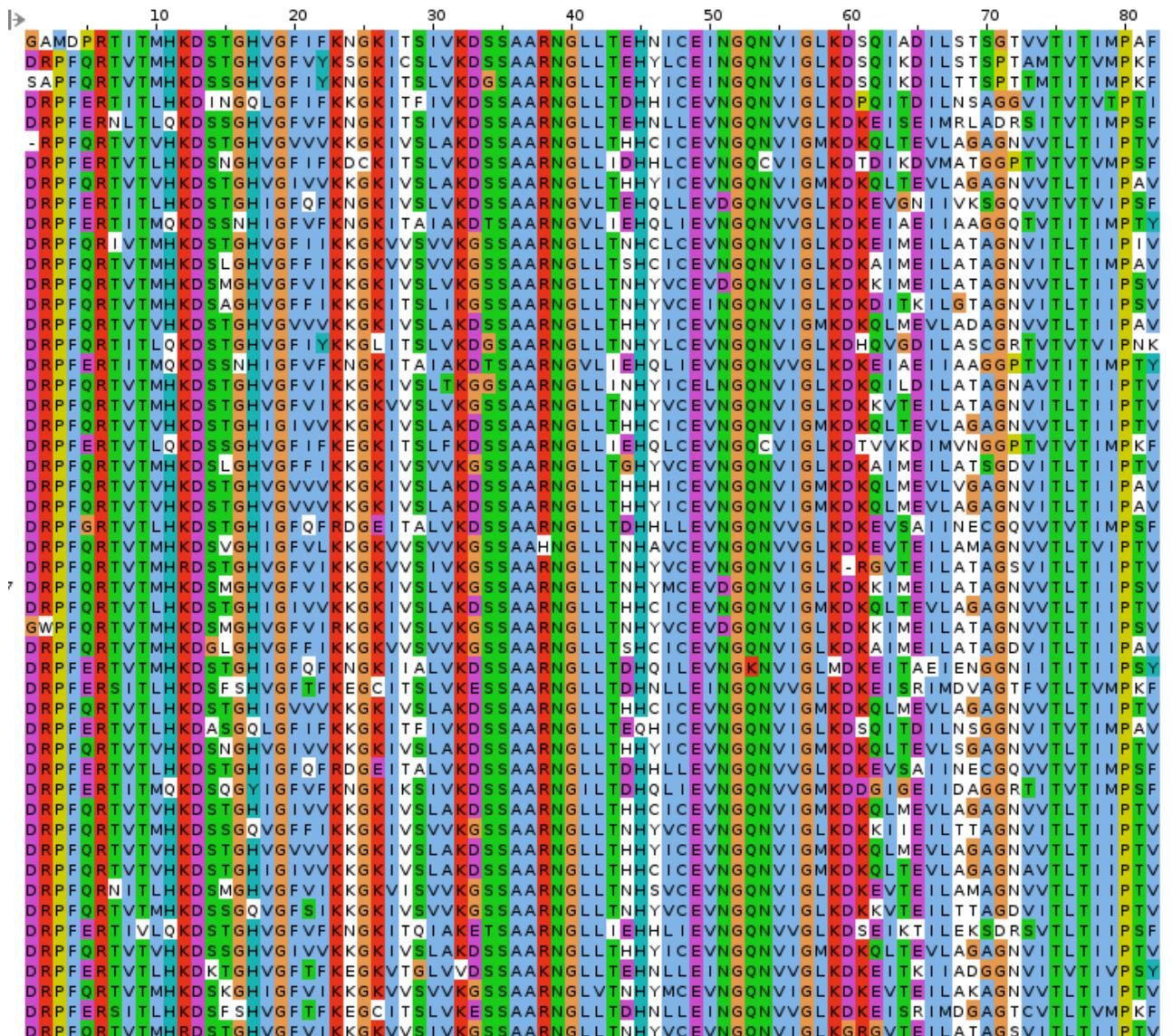


Figure 4.6 – L'alignement de notre sélection de séquences homologues à la protéine Syntenin (code PDB 1R6J)



Figure 4.7 – Une sélection de séquences protéiques, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine Syntenin (code PDB :1R6J), modèle NEA

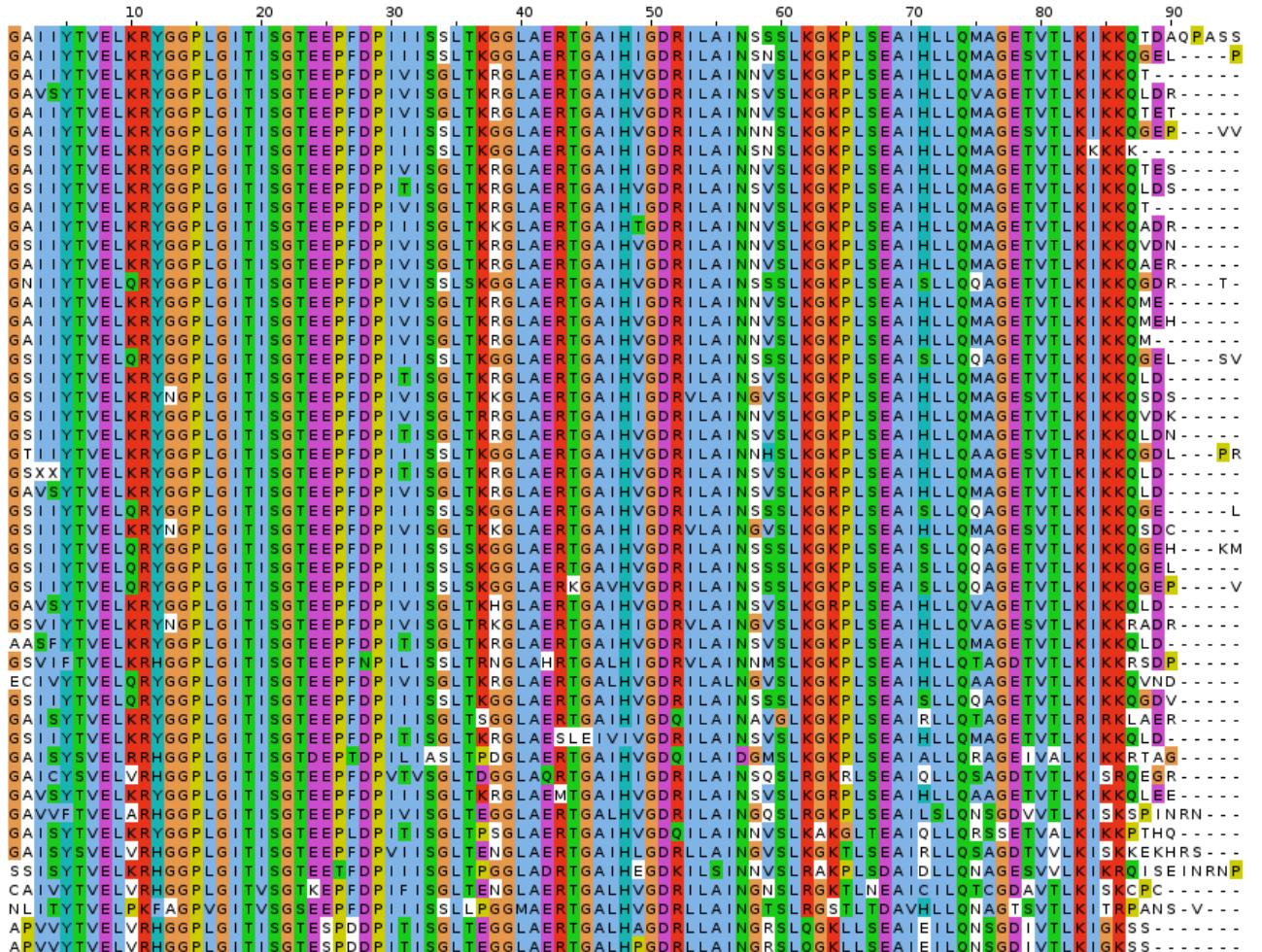


Figure 4.8 – L’alignement de notre sélection de séquences homologues à la protéine GRIP (code PDB : 1N7E)

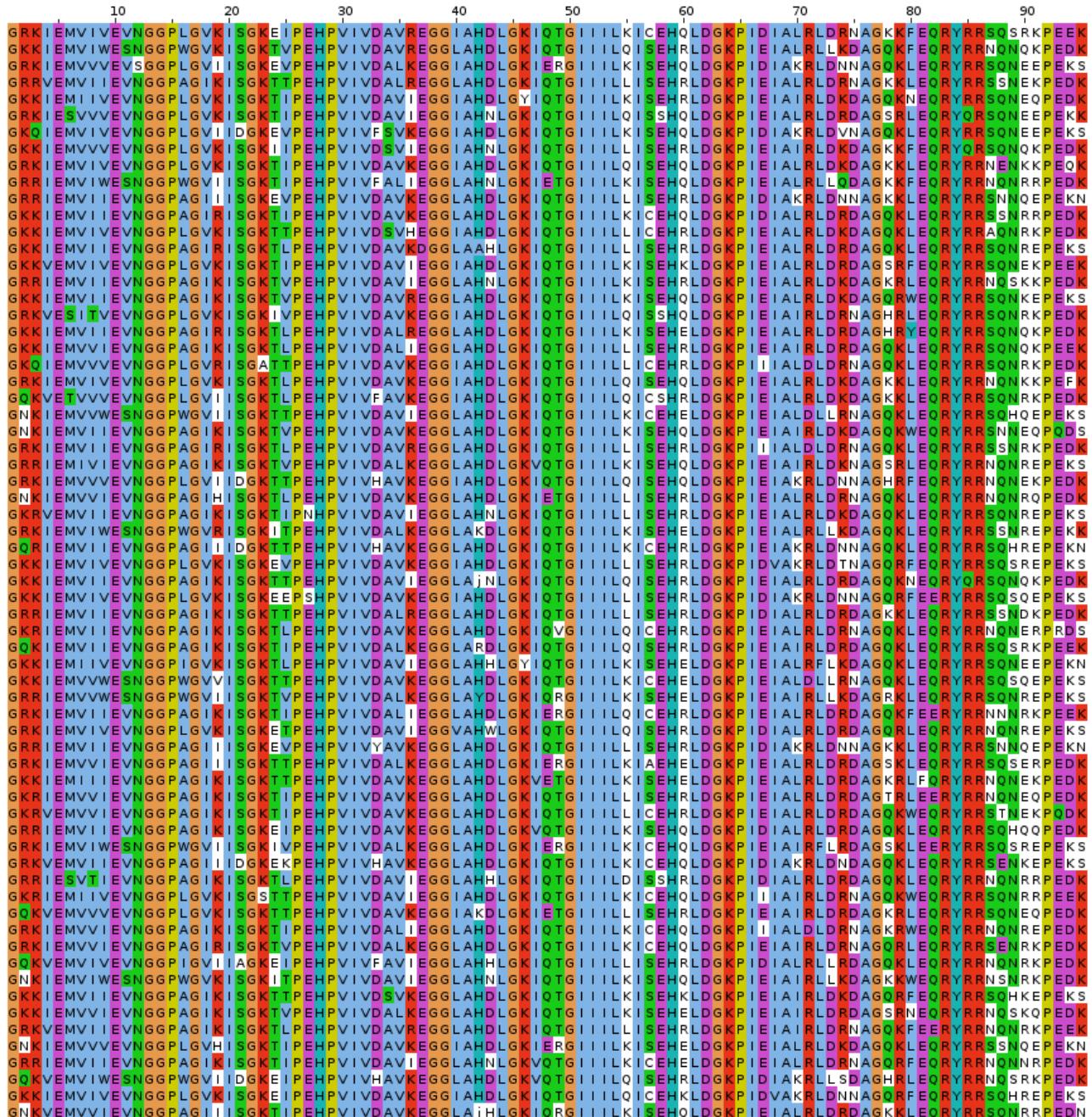


Figure 4.9 – Une sélection de séquences protéins, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine GRIP (code PDB :1N7E), modèle NEA

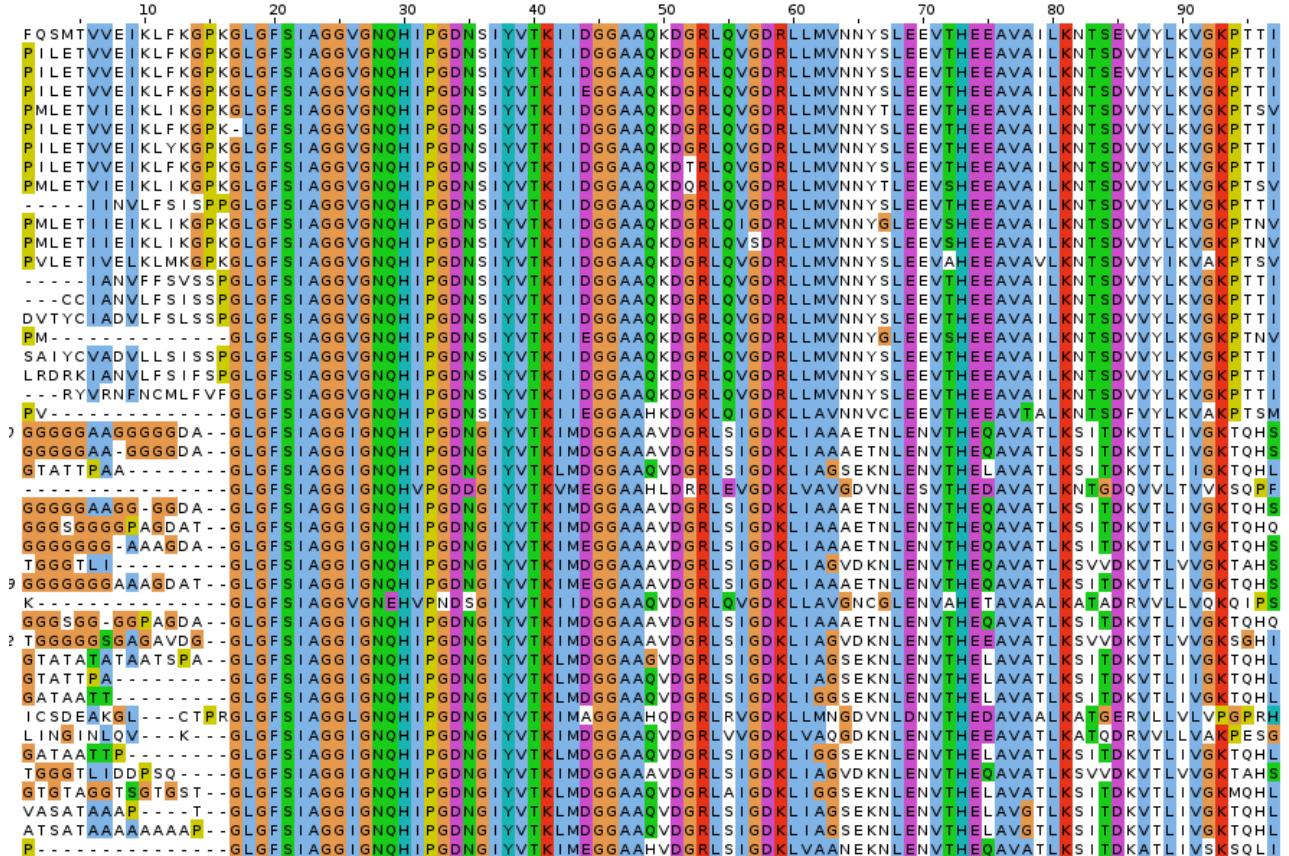


Figure 4.10 – L’alignement de notre sélection de séquences homologues à la protéine DLG2 (code PDB 2BYG)

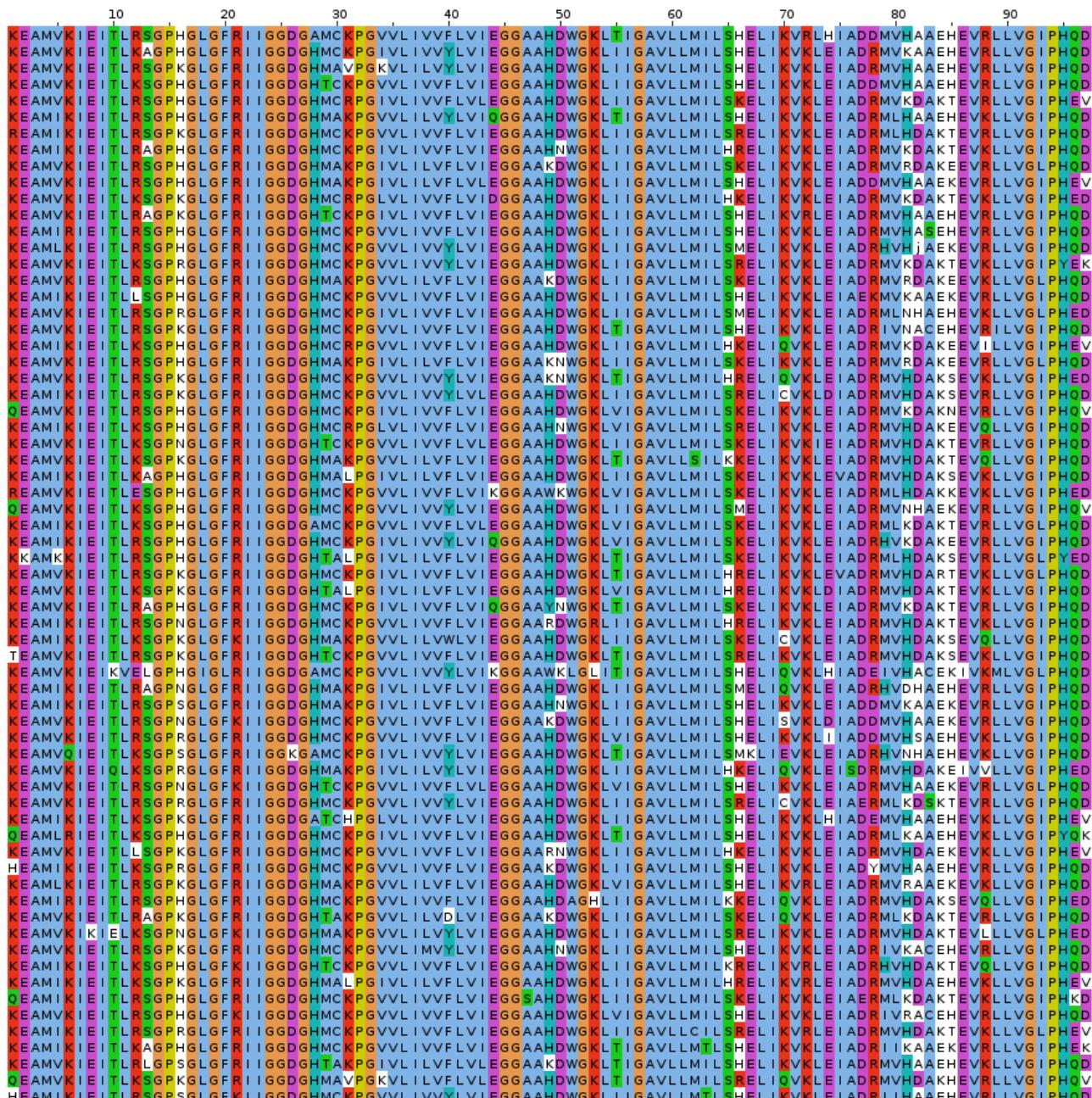


Figure 4.11 – Une sélection de séquences protéiques, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine DLG2 (code PDB 2BYG), modèle NEA

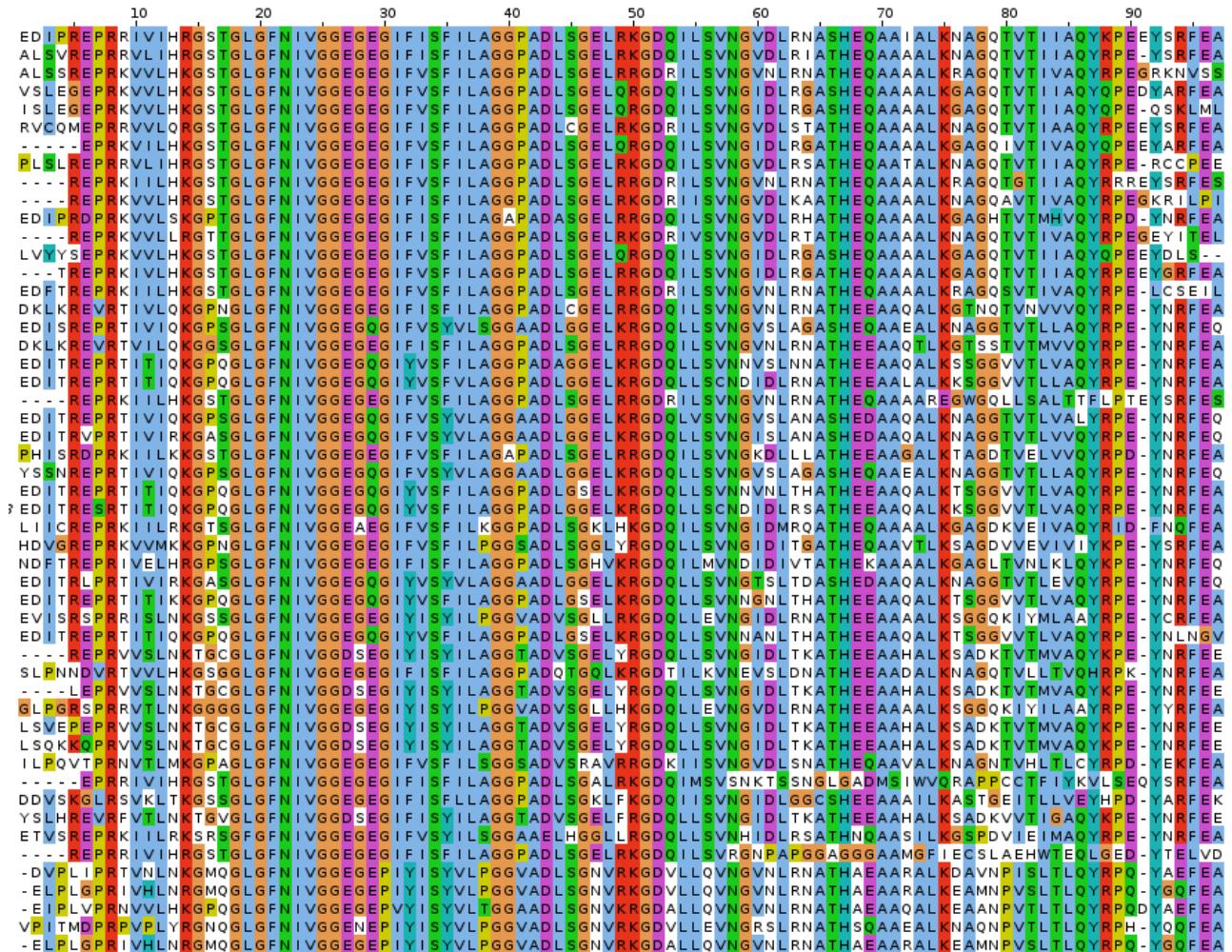


Figure 4.12 – L’alignement de notre sélection de séquences homologues à la protéine PSD95 (code PDB 3K82)

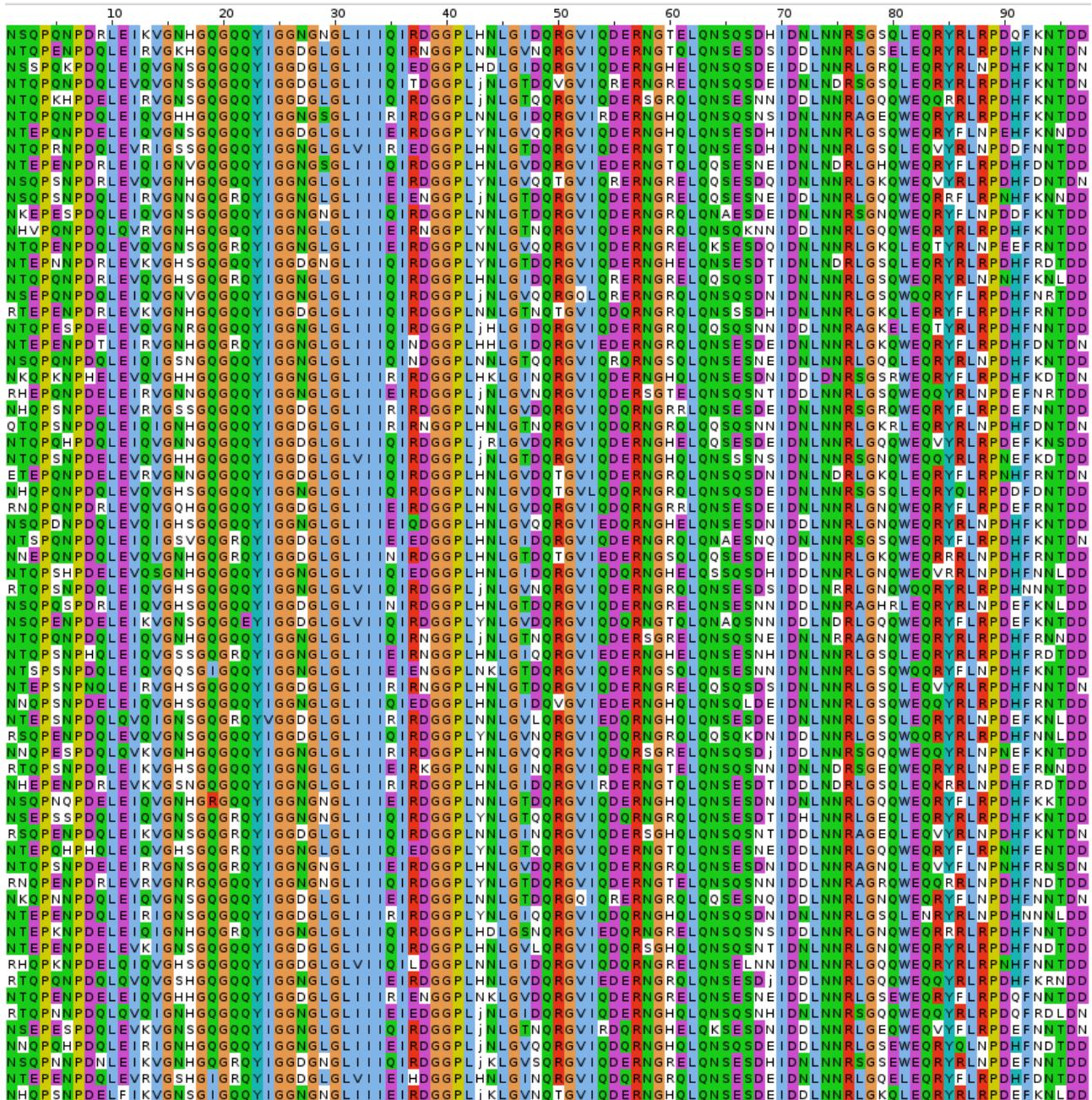


Figure 4.13 – Une sélection de séquences protéins, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine PSD95 (code PDB :3K82), modèle NEA

4.6 Séquences calculées

Pour réaliser les calculs Monte-Carlo (MC), les structures ont été préparées et les matrices d'énergie calculées à l'aide d'une procédure décrite précédemment ?? . Deux segments manquants dans le domaine Tiam1 (résidus 851-854 et 868-869) ont été construits en utilisant le programme Modeller (?). Le ligand peptidique a été retiré de la structure PDB avant de calculer la matrice d'énergie. Pour chaque paire d'acides aminés, l'énergie d'interaction a été obtenue après 15 pas de minimisation de l'énergie, avec le backbone fixé, seules les interactions de la paire entre les autres chaînes et le backbone sont prises en compte. Cette courte minimisation simplifie l'approximation discrète. Les rotamères de chaînes latérales utilisés sont une version légèrement étendue de la librairie de Tuffery et col ?, qui possède un total de 254 rotamères (sur l'ensemble des types d'acides aminés). Cette extension comprend des orientations d'hydrogène supplémentaires pour les groupes OH et SH ?. Cette bibliothèque de rotamères a été choisie pour sa simplicité et parce qu'elle a donné de très bonnes performances dans les tests de placement de chaînes en comparaison au programme spécialisé scwlr4 qui utilise une bibliothèque beaucoup plus grande ??.

4.6.1 simulation Monte-Carlo

La conception des séquences est réalisée avec Proteus ?. Pour optimiser les énergies de références, les positions dans lesquels la séquence native comporte une Glycine ou une Proline conservent toujours leur type naturel et les positions mutables ne peuvent pas le faire en Gly et Pro. Pour optimiser les énergies de référence. Nous sélectionnons, alternativement le long du backbone, une position pouvant muter, une position ne pouvant pas muter. Dans un second temps, toutes les positions sont libres de muter (sauf Gly et Pro). Ensuite, les modèles optimisés sont testés avec des simulations où ce dernier ensemble de positions mutables est utilisé. Dans tous les cas, des mutations sont produites au hasard, soumises uniquement à la fonction d'énergie MMGBSA qui entraîne la simulation. Les simulations Monte-Carlo utilisent des mouvements à une ou deux positions, où les rotamères, les types d'acides aminés ou les deux peuvent changer. Pour les mouvements à deux positions, la deuxième position est choisie parmi celles qui ont une énergie d'interaction significative avec la première pour au moins une conformation du couple formé de leur chaîne latérale respective (c'est à dire 10 kcal/mol ou plus). De plus, l'échantillonnage est amélioré par l'échange de répliques (REMC), où plusieurs simulations MC sont exécutées en parallèle, à différentes températures. Des échanges périodiques sont effectués comme indiqué dans ??.

4.6.2 Génération de séquence Rosetta

Des simulations Monte-Carlo sont également réalisées à l'aide du programme et de la fonction d'énergie Rosetta ?. Les simulations sont faites en utilisant la version 2015.38.58158 de la suite (librement disponible en ligne), en utilisant la commande :

```
fixbb -s Tiam1.pdb -resfile Tiam1.res -nstruct 10000 -ex1 -ex2 -linmem_ig 10
```

où les options ex1 et ex2 activent une recherche améliorée des rotamères pour les chaînes latérales enfouis. La dernière option correspond au calcul de l'énergie au cours de la recherche MC, et les paramètres par défaut sont utilisés pour les autres options. Comme pour les simulations Proteus, les résidus Gly et Pro présents dans la protéine sauvage ne sont pas autorisé à muter, et les positions qui mutent ne peuvent pas le faire en Gly ou Pro. Des simulations sont exécutées pour chacun de nos domaines PDZ jusqu'à obtenir 10 000 séquences uniques de faible énergie, ce qui correspond à des durées d'exécution d'environ 5 minutes par séquence sur un seul cœur d'un processeur Intel récent, pour un total de 10 heures par protéines en utilisant 80 cœurs. C'est tout à fait comparable au coût des calculs Proteus, en comptant le temps de calcul de la matrice d'énergie plus celui des simulations Monte-Carlo.

4.6.3 Caractérisation des séquences calculées

Les séquences calculées sont comparées à l'alignement Pfam pour la famille PDZ, en utilisant la matrice Blosum40 et une pénalité d'écart de -6. Cette matrice est appropriée pour comparer des homologies éloignées (les séquences CPD et celles de Pfam). Chaque séquence Pfam est également comparée à l'alignement Pfam, ce qui permet de comparer des séquences calculées et un couple de domaines PDZ naturels. Pour ces comparaisons Pfam/Pfam, si un domaine de test T fait partie de l'alignement, la comparaison T/T n'est pas prise en compte, pour être plus cohérent avec les comparaisons calculées/Pfam. L'alignement Pfam utilisée est le « RP55 » (voir 4.4.4). Les similitudes sont calculées pour les 14 résidus du cœur et pour l'ensemble des positions mutables de la protéine.

Les séquences calculées sont soumises à la bibliothèque de modèle de Markov Caché Superfamily ?? qui tente de classer les séquences selon la base de données structurelle de protéines SCOP ?, voir 4.4.1. Le programme hmmscan est exécuté avec un seuil de valeur E de 10^{-10} .

4.7 Résultats du modèle NEA

4.7.1 optimisation du modèle de l'état déplié

Nous optimisons les énergies de référence E_t^r pour les six protéines, en utilisant leurs homologues naturels pour définir les fréquences d'acides aminés cibles. La constante diélectrique de la protéine ϵ_p est égale 8. La méthode d'optimisation utilisée est la méthode linéaire voir (4.2.3) avec 20 itérations avec la contrainte des groupes et 20 itérations sans cette contrainte,l'optimisation ne se fait plus alors sur les six classes de type d'acide aminé , mais directement sur les dix-huit types possibles. La fonction Proxy calculée sur les groupes de types converge autour les valeurs 0.06, 0.07 (pour respectivement les énergies enfouies et exposées) et pour les types 0.08 et 0.14 pour les groupes. Ce qui correspond à des variations pour les E_t^r inférieurs à $0,05\text{Kcal/mol}$ pour tous les types d'acides aminés sur les 5 derniers cycles d'optimisation. Le tableau 4.11 les donne énergies de référence finales, qui sont utilisées ensuite pour la génération de séquences.

Table 4.6 – Les énergies de référence obtenues avec l'optimisation 6 protéines.

Type d'acides aminés	Pos. Enf.	Pos Exp.
ALA	0,00	0,00
ARG	-28,29	-28,90
ASN	-5,94	-6,00
ASP	-9,19	-9,80
CYS	-1,04	-1,04
GLN	-4,72	-4,78
GLU	-7,90	-8,51
HID	11,96	12,39
HIE	11,43	11,85
HIP	14,53	14,96
ILE	4,72	2,11
LEU	1,17	-1,44
LYS	-4,56	-4,47
MET	-2,78	-3,54
PHE	-0,37	-2,55
SER	-3,73	-2,80
THR	-3,82	-3,82
TRP	-1,61	-3,79
TYR	-4,20	-6,10
VAL	0,83	-1,77

Le tableau 4.7 compare les fréquences d'acide aminé des homologues naturels et les calculées. La population théorique des différentes classes d'acide aminé a bien rejoint

Table 4.7 – Les compositions en acide aminé (%) des séquences expérimentales et Proteus après optimisation des E_t^s . Les différences entre expérimentale et théorique sont indiquées entre parenthèses.

Res	Experimentale				Proteus		
	Enfoui		Exposé		Enfoui		Exposé
ALA	10.9		4,6		11,1	17,0	4,4
CYS	1.3	16,9	0.5	13,4	0.0	(0.1)	0.3
THR	4.7		8.3		5.9		7.3
ASP	4.3	6,8	6,0	17,9	4,5	6,7	5,6
GLU	2.5		11.9		2.2	(-0,1)	11,1
ASN	2.6	4,7	6,7	12,2	2,5	4,7	7,5
GLN	2.1		5.5		2.2	(0,0)	6,5
HIP	1.2		5,0		1,0		5,2
HIE	0.0	1,2	0.0	5,0	0.1	(-0,1)	0.4
HID	0.0		0.0		0.0		(0.6)
ILE	16.0		4.2		16.9		4.1
VAL	16.5	50.7	5.4	14.0	16.7	52.1	5.6
LEU	18.2		4.4		18.5	(1.4)	4.3
LYS	2,5	2,5	10,9	10,9	1,5	1,5 (-1,0)	13,0 (2.1)
MET	0,9	0,9	1,5	1,5	1,6	1,6 (0,7)	1,4 (-0.1)
ARG	2,8	2,8	8,7	8,7	2,5	2,5 (-0,3)	6,1 (-2.6)
SER	5,3	5,3	7,6	7,6	4,3	4,3 (-1,0)	8,7 (1.1)
PHE	4,1	4,1	2,4	2,4	4,5	4,6	2,1
TRP	0.0		0.0		0.1	(0,5)	0,0
TYR	2,6	2,6	1,2	1,2	2,2	2,2 (-0,4)	0,4 (-0.8)
GLY	0.8	0,9	3,1	4,9	0,0	0,0	0,0
PRO	0.1		1.8		0.0	(-0,9)	0,0
							(-4,9)

Les types d'acides aminés sont

rassemblés selon les groupes d'optimisations.

l'expérience, avec des écarts de moins de 1 dans la majorité des cas, pour les positions exposées et pour les positions enfouies, seulement deux groupes ont des écarts de plus de 2 (2,1 et 2,6 pour Lys et Arg positions exposées). L'accord entre les types d'acide aminé est aussi bon avec au pire des cas les mêmes écarts à plus de 2 que pour les groupes de type. Pendant les 20 premiers cycles d'optimisation, la distribution des fréquences intra classe dépend par construction du δE_t^r défini dans pour chaque classe, qui ont été calculés avec la mécanique moléculaire (voir méthodes). Mais la seconde série de 20 itérations permet l'ajustement de ces valeurs.

4.7.2 Tests de reconnaissance de famille

À partir, nos énergies optimisées, nous générerons des séquences pour chaque protéine. Les simulations Proteus utilisent l'algorithme Monte-Carlo avec échange de répliques (REMC) avec huit répliques (ou marcheurs) et 750 millions de pas par réplique, avec des énergies thermiques kT qui varient de 0,125 à 3 kcal/mol. Toutes les positions sont autorisées à muter librement dans tous les types d'acides aminés exceptés Gly et Pro. Les simulations ont été faites avec la fonction d'énergie MMGBSA, sans aucune introduction de biais vers les séquences naturelles ni aucune limite sur le nombre de mutations. Les 10 000 séquences avec les énergies les plus faibles parmi celles échantillonnées par au moins une des répliques MC sont retenues pour l'analyse. De la même façon, 10 000 séquences produites par Rosetta sont retenues. Ces séquences sont analysées par les outils de reconnaissance de repliement « Superfamily » (voir ??). Avec une constante diélectrique de 8, nous avons obtenu un pourcentage élevé de séquences correctement associées à la famille et superfamille PDZ : 100% pour NHREF, INAD, GRIP, DLG2, 99% pour Syntenin, seule PSD95 donne un score assez mauvais de 47% pour la famille et 50% pour la superfamille. Les E-values sont inférieures à 10^{-3} pour les affectations à la famille. Ces valeurs sont semblables à celles obtenues par Rosetta pour les cinq premiers cas, par contre l'affectation à la superfamille meilleure pour Rosetta avec cas E-values compris entre $3.7 \cdot 10^{-23}$ et $1.3 \cdot 10^{-9}$, alors que Proteus obtient des E-values compris entre $4 \cdot 10^{-4}$ et $2 \cdot 10^{-12}$ si l'on exclut PSD95 les détails sont présentés aux tables 4.8 et 4.9.

Protein	Match/seq size	Superfamily Evalue	Superfamily success	Family Evalue	Family success
NHREF	81/91	$2.00 \cdot 10^{-12}$	10000	$9.97 \cdot 10^{-3}$	10000
INAD	84/94	$4.80 \cdot 10^{-11}$	10000	$2.83 \cdot 10^{-3}$	10000
GRIP	82/95	$4.73 \cdot 10^{-8}$	10000	$5.56 \cdot 10^{-3}$	10000
Syntenin	63/91	$4.01 \cdot 10^{-4}$	9999	$1.05 \cdot 10^{-2}$	9999
DLG2	84/97	$3.82 \cdot 10^{-10}$	10000	$3.75 \cdot 10^{-3}$	10000
PSD95	46/97	$7.65 \cdot 10^{-1}$	5029	$4.06 \cdot 10^{-2}$	4719

Table 4.8 – Résultats Superfamily pour les séquences Proteus avec le modèle NEA.

4.7.3 Séquences et diversité de séquences

Une sélection des meilleures séquences calculées par Proteus, au sens de l'énergie, pour le sous-ensemble de 6 protéines est montrée aux figures 4.3 4.54.74.94.114.13 et les homologues naturels aux figures 4.2 4.44.64.84.104.12 (la coloration est celle de clustalX

Protein	Match/seq size	Superfamily Evalue	Superfamily success	Family Evalue	Family success
NHREF	79/91	$1.3 \cdot 10^{-13}$	10000	$2.2 \cdot 10^{-3}$	10000
INAD	85/94	$7.4 \cdot 10^{-14}$	10000	$3.7 \cdot 10^{-3}$	10000
GRIP	84/95	$2.2 \cdot 10^{-10}$	10000	$1.2 \cdot 10^{-3}$	10000
Syntenin	76/82	$7.3 \cdot 10^{-13}$	10 000	$1.8 \cdot 10^{-3}$	10 000
DLG2	86/97	$1.3 \cdot 10^{-9}$	10 000	$9.6 \cdot 10^{-4}$	10 000
PSD95	90/97	$3.7 \cdot 10^{-23}$	10 000	$5.2 \cdot 10^{-4}$	10 000

Table 4.9 – Résultats Superfamily pour les séquences Rosetta

implémenté dans Jalview cite{Jalview}). Comme on l'a vu dans de nombreuses études de CPD antérieures (30,72) l'accord avec l'expérience pour les positions du cœur est très bon, alors que l'accord en ce qui concerne les résidus de surface est nettement plus faible. La diversité des séquences naturelles et des séquences calculées est caractérisée par la moyenne sur la séquence de l'exponentielle de l'entropie résiduelle (voir 4.4.7). Comme référence nous utilisons l'ensemble Pfam Seed qui est constitué d'un sous-ensemble représentatif des domaines PDZ naturels (refsubsection :AlignPfam). L'entropie est calculée aux positions de chaque protéine. La diversité des séquences Rosetta est légèrement meilleure avec des valeurs comprises entre 1.4 et 1.68 alors que celles de Proteus sont comprises 1.24 et 1.55 tandis que la diversité de Pfam Seed se situe à une entropie moyenne de 3,11. Le regroupement des séquences de NHREF, INAD, GRIP, Syntenin, DLG2 et PSD95 calculés donne une entropie de 2,88 avec Rosetta et 2,42 avec Proteus. Ce qui indique que ces six seules géométries de backbone ne peuvent pas atteindre les mêmes niveaux de diversités que l'ensemble Seed conçu pour caractériser la diversité des domaines PDZ et notamment la diversité de leur backbone.

4.7.4 Scores de similarité Blosum

Les scores de similarité Blosum40 entre les séquences calculées et les séquences naturelles sur l'ensemble des positions sont globalement faibles (voir la figure 4.14). les similitudes Proteus et Rosetta se chevauchent au pied du sommet des scores naturels pour NHREF, INAD, GRIP et DLG2 avec des valeurs Proteus en retrait par rapport à celles de Rosetta de quelques dixièmes de points, moins de 20 pour NHREF, mais près de cinquante pour Syntenin. Les résultats Proteus pour PSD95 sont beaucoup moins bons que ceux de Rosetta avec un écart moyen de plus de 70. Pour les résidus du cœur, montrés à la figure , 4.15 la similitude des séquences calculées avec les séquences naturelles est beaucoup plus forte, avec beaucoup de séquences Proteus avec des scores à plus de 30 pour NHREF, INAD

Protein	Proteus	Rosetta	Pfam seed
NHREF	1.38	1.45	3.15
INAD	1.37	1.55	3.06
GRIP	1.33	1.44	3.09
Syntenin	1.39	1.43	3.03
DLG2	1,24	1,57	3,11
PSD95	1,27	1,40	3,15
6prots	2,42	2,88	
CASK	1.55	1.68	3.15
TIAM1	1,22	1,57	3,15

Table 4.10 – Moyenne de l'exponentielle de l'entropie sur les ensembles des positions des protéines

et DLG2. Proteus fait jeu égal avec Rosetta sur NHREF et Syntenin, et est globalement meilleur sur INAD et DLG2, et moins bon sur GRIP et PSD95.

4.7.5 Tests de validation croisée

Cette partie a été effectuée en commun avec Nicolas Panel, doctorant de notre laboratoire. Les détails sont publiés dans ?. Comme premier test de validation croisée, nous appliquons nos énergies de référence aux deux domaines, notre sous-ensemble S_2 , qui ne font pas partie du sous-ensemble S_1 utilisé pendant l'optimisation, voir 4.5.6. Ainsi, nous générerons des séquences Tiam1 et Cask qui sont alors soumises aux tests Superfamily et aux calculs de similarité. La performance de Tiam1 sur la superfamille ,84,6%, est un peu en dessous de celles obtenues sur les autres protéines. Le score de Tiam1 pour la reconnaissance de la famille est à 76,6%. Les scores de Cask sont du même ordre que ceux de nos 6 premières protéines avec 100% de reconnaissance pour la famille et la superfamille, avec des E-value comparables.

Comme validation croisée supplémentaire, les énergies de références ont été optimisées par Nicolas Panel, en utilisant notre sous-ensemble S_2 comme ensemble alternatif de domaines PDZ, et la troisième méthode d'optimisation, voir 4.12. Nous générerons alors, avec ce modèle, des séquences pour Syntenin et DLG2. Encore une fois, les scores sont proches de ceux obtenus avec le modèle optimisé sur 6 protéines ; les reconnaissances sont à 100% et les E-values montent de $4.01 \cdot 10^{-4}$ à $1.3 \cdot 10^{-2}$ avec syntenin pour la superfamille et de $3.82 \cdot 10^{-10}$ à $8.0 \cdot 10^{-9}$ pour DLG2.

Les histogrammes des scores de similarité Blosum montrent que les scores globaux pour Tiam1 et Cask sont très semblables pour les deux modèles. Pour DLG2 et Syntenine,

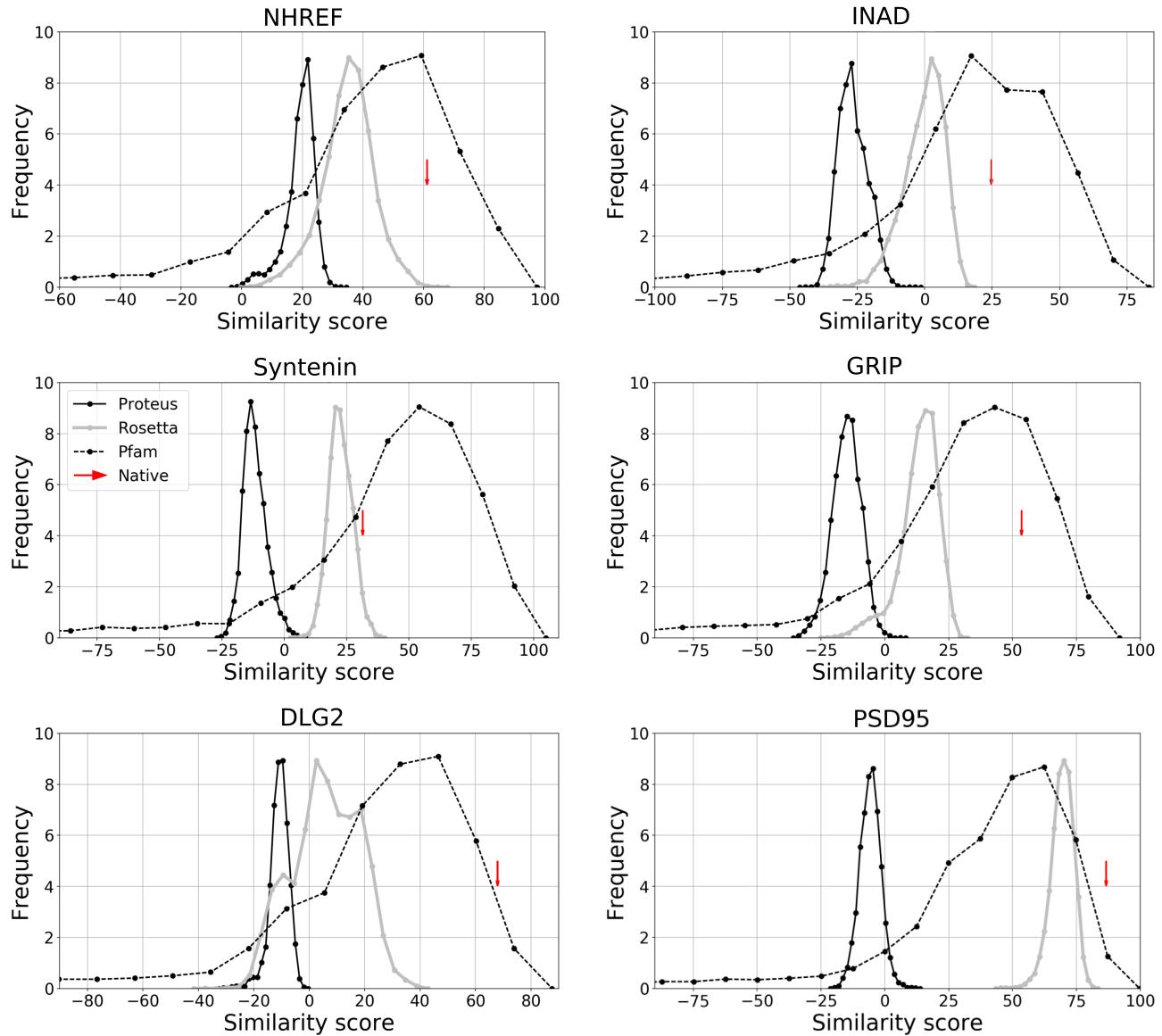


Figure 4.14 – Similarité des séquences des 6 protéines produites par Proteus et Rosetta à l’alignement Pfam RP55, sur l’ensemble des positions.

nous calculons également les scores de similarité en utilisant les deux modèles. Les scores de similarité avec le modèle S_2 sont légèrement plus faibles qu’avec le modèle S_1 . Le score global a diminué d’environ 20 points pour la Synténine et environ 10 points pour DLG2. dans l’ensemble, les modèles de validation croisée ont légèrement dégradé les performances. Ainsi, pour tout domaine d’intérêt PDZ, il est préférable d’optimiser les énergies de référence spécifiquement pour ce domaine plutôt que de transférer des valeurs paramétrées en utilisant d’autres domaines PDZ.

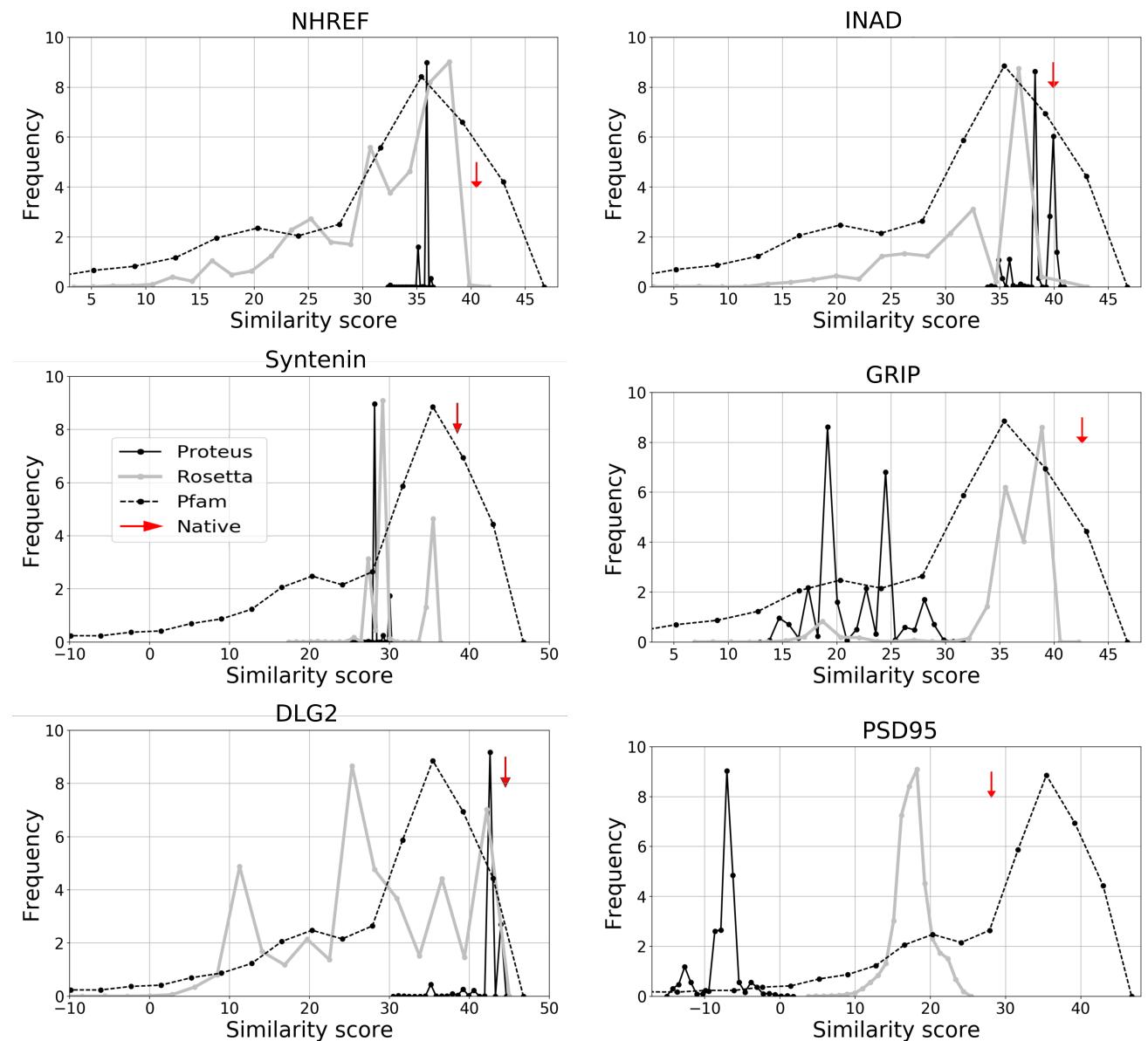


Figure 4.15 – Similarité des séquences des 6 protéines produites par proteus modèle NEA et Rosetta à l’alignement Pfam RP55, sur les positions du cœur hydrophobe.

4.8 Résultats du modèle FDB

4.8.1 optimisation du modèle de l’état déplié

Nous optimisons maintenant les énergies de référence E_t^r pour trois protéines : NHREF, Syntenin et DLG2. Ici, la constante diélectrique de la protéine est fixée à 4. La méthode d’optimisation utilisée est la méthode parabolique voir (4.2.3) avec 20 itérations avec la contrainte des groupes de type d’acides aminés et 20 itérations sans cette contrainte.

Chaque itération se fait avec 100 millions de pas. La fonction Proxy converge aux valeurs 0.06 et 0.03 , pour les énergies enfuiées et respectivement les énergies exposées et aux valeurs 0.03 (enfuis) et 0.02 (exposés) pour la fonction proxy calculées sur les types.Ce qui donne pour les E_t^r , les variations inférieures à $0,05\text{Kcal/mol}$ sur les 5 derniers cycles , pour tous les types, sauf pour Arg et Hip dans le cas enfui, où les variations montent à $0,1\text{Kcal/mol}$.La population des types d'acide aminé est proche de la population sauvage.Les écarts sur les groupes de types dans le cas des positions enfuiées sont inférieurs à 1% sauf pour le groupe Asp,Glu 1,2% et Hip,Hie,Hid 1,4%. Dans le cas exposé, les écarts sont un peu moins bons avec 5 groupes entre 2 et 3% d'écart, sachant que les Gly et Pro ne sont pas générés par Proteus.Les détails sont donnés dans le tableau ??.

Pour pouvoir évaluer l'apport de l'optimisation FDB dans notre modèle, nous optimisons également les E_t^r pour les trois protéines avec le modèle NEA et la constante diélectrique de la protéine à 4. La même méthode est utilisée.Dans ces conditions,les E_t^r se stabilisent à 0.05 pour tous les types enfuis ou exposés.Ces 4 jeux d'énergies sont donnés dans le tableau 4.12.

4.8.2 Tests de reconnaissance de famille

À présent, nous générerons des séquences pour chaque protéine et pour chacun des jeux des E_t^r FDB et NEA. Le protocole pour Proteus est identique à celui utilisé voir de la section 4.7,la constante diélectrique de la protéine étant ici de 4. Ici encore, les 10 000 séquences de meilleures énergies parmi celles échantillonnées par les répliques MC sont retenues pour l'analyse.Les résultats Superfamily sont présentés au tableau 4.13. Pour le FDB, la reconnaissance des familles et des superfamilles est de 100% pour les 3 protéines tout comme Rosetta. En termes de E-value, Proteus FDB fait jeu égal avec Rosetta, avec des valeurs pour la super-famille allant de $2.85 \cdot 10^{-6}$ jusqu'à $8.54 \cdot 10^{-14}$ pour Proteus et allant de $1.3 \cdot 10^{-9}$ à $1.3 \cdot 10^{-13}$ pour Rosetta et des valeurs très proches pour la famille.Pour le NEA, les résultats sont corrects pour la reconnaissance avec 98% pour les trois protéines, mais nettement moins bons pour les E-value de la Superfamille.

4.8.3 Scores de similarité Blosum

Les scores de similarité Blosum40 sont calculés entre les séquences conçues par informatique et les séquences Pfam.Nous calculons ces similarités pour les séquences proteus FDB, proteus NEA et Rosetta, sur l'ensemble des positions moins une partie au début et une partie à la fin de la séquence, parce qu'elles ne sont pas conservées dans l'alignement Pfam. Cela représente moins de 10% de la longueur de la séquence.Pour NHREF,l'approximation

Table 4.11 – La composition en acide aminé (%) des séquences expérimentales et Proteus après optimisation des énergies de référence. La différence entre expérimentales et Proteus sur les classes est donnée entre crochets.

Res	3 protéines expérimentales				FDB			
	Enfoui		Exposé		Enfoui		Exposé	
	type	classe	type	classe	type	classe	type	classe
ALA	8,7		5,5		9,6		1,8	
CYS	1,9	16,8	0,4	13,6	2,8	[−0,3]	0,6	[2,3]
THR	6,2		7,7		4,7		8,9	
SER	4,4	4,4	6,7	6,7	5,2	[−0,8]	7,9	[−1,2]
ASP	4,8		6,1		5,8		8,1	20,6
GLU	2,6	7,4	11,0	17,1	2,8	[−1,2]	12,5	[−3,5]
ASN	3,4		6,9		4,2		8,8	16,0
GLN	1,7	5,1	5,8	12,7	1,8	[−0,9]	7,2	[−3,3]
HIP	2,0		5,9		0,0		0,4	
HIE	0,0	2,0	0,0	5,9	0,5	[1,4]	2,7	[0,9]
HID	0,0		0,0		0,1		1,9	
ILE	12,4		4,1		11,2		0,6	
VAL	21,1	50,3	5,1	14,0	21,2	[0,9]	5,2	12,5
LEU	16,8		4,8		17,0		6,7	[1,5]
MET	1,3	1,3	1,8	1,8	1,0	[0,3]	2,2	[−0,4]
LYS	3,4	3,4	10,8	10,8	4,2	[−0,8]	12,4	[−1,6]
ARG	1,1	1,1	9,8	9,8	1,8	[−0,7]	11,8	[−2,0]
PHE	4,6		2,4		3,9		0,0	
TRP	0,0	4,6	0,1	2,5	0,0	[0,7]	0,0	[2,5]
TYR	2,4	2,4	1,2	1,2	2,0	[0,4]	0,0	[1,2]
GLY	1,0		1,9		0,0		0,0	
PRO	0,1	1,1	1,8	3,7	0,0	[1,1]	0,0	[3,7]

Table 4.12 – Les énergies de référence obtenues avec l'optimisation sur 3 protéines.

acides aminés	NEA		FDB	
	Pos.	Enf.	Pos	Exp.
ALA	0,00	0,00	0,00	0,00
CYS	-0,89	-2,57	-1,06	-1,64
THR	-5,31	-8,075	-4,84	-6,68
SER	-5,55	-6,55	-4,45	-5,24
ASP	-17,26	-22,06	-14,56	-18,82
GLU	-16,12	-20,68	-14,52	-18,21
ASN	-16,38	-20,41	-14,02	-17,80
GLN	-14,00	-18,41	-13,14	-16,61
HID	11,21	6,95	10,85	8,13
HIE	10,63	6,15	10,41	7,37
HIP	15,17	10,72	12,86	10,98
ARG	-53,40	-57,36	-51,37	-54,76
LYS	-8,20	-12,34	-8,24	-11,35
ILE	6,76	3,44	5,50	3,06
VAL	0,43	-2,19	-0,05	-1,66
LEU	0,52	-3,72	0,00	-2,94
MET	-1,61	-3,21	-2,85	-3,09
PHE	1,86	-2,68	0,17	-3,18
TRP	-0,23	-7,67	-1,94	-5,53
TYR	-5,10	-10,90	-5,91	-10,14

FDB améliore très nettement les scores de Proteus NEA avec un écart d'environ 50. Ce qui place Proteus à peu près au niveau de Rosetta. Pour Syntenin, les écarts sont plus serrés avec le FDB légèrement moins bon que le NEA, mais qu'en même proche de Rosetta. Dans le cas de DLG2, le FDB domine les séquences NEA et près de la moitié de celles produites par Rosetta. Sur les positions du cœur hydrophobe, Les résultats Proteus sont excellent, avec le plus souvent, des similarités avec Pfam compris entre 30 et 40. Le NEA et le FDB font jeu égal sur DLG2, le NEA étant au-dessus pour les deux autres. Il n'y a donc pas d'amélioration sur le cœur ce qui s'explique par le fait que ces résidus sont trop éloignés du solvant pour bénéficier de l'effet de l'approximation FDB. Les scores Rosetta pour Syntenin sont proches, mais pour les deux autres, les scores nettement plus variables et en globalement moins bons. Tout cela est représenté à la figure ??.

Model	Protein size	Match/seq E-value	Superfamily success	Superfamily E-value	Family success	Family
Proteus FDB epsilon=4	NHREF	80/91	8.54 10 ⁻¹⁴	10000	8.94 10 ⁻³	10000
	Syntenin	70/82	2.85 10 ⁻⁶	10000	2.69 10 ⁻³	10000
	DLG2	88/97	3.26 10 ⁻¹²	10000	1.96 10 ⁻³	10000
Rosetta	NHREF	79/91	1.3 10 ⁻¹³	10000	2.2 10 ⁻³	10000
	Syntenin	76/82	7.3 10 ⁻¹³	10000	1.8 10 ⁻³	10000
	DLG2	86/97	1,3 10 ⁻⁹	10 000	9,6 10 ⁻⁴	10 000
Proteus NEA epsilon=4	NHREF	62/91	3,22 10 ⁻³	9857	1,00 10 ⁻²	9857
	Syntenin	70/82	2.83 10 ⁻³	9879	3,62 10 ⁻³	9879
	DLG2	83/97	1,66 10 ⁻³	9876	3,18 10 ⁻³	9876

Table 4.13 – Résultats Superfamily pour les séquences Proteus avec le modèle FDB (énergies de références optimisées sur 20 cycles selon les classes + 20 cycles selon les types).

4.8.4 Taux d'identité à la séquence native

Pour chaque protéine, nous calculons le taux d'identité des 10 000 séquences de meilleures énergies par rapport à la séquence native, ainsi que pour les 10 000 séquences Rosetta ,voir 4.4.3.On considère uniquement les positions mutables sans celles aux extrémités des séquences comme expliqué au paragraphe précédent.Ce taux varie pour Proteus FDB entre 24% et 33% , et entre 20% et 33% pour la version NEA. Pour Rosetta les taux se situent entre 35 et 40% avec un écart de 11% pour NHREF sur le FDB et de 7% pour les deux autres protéines. voir la table 4.14. Il s'avère donc que les séquences Rosetta sont nettement plus proches des séquences natives que celles de Proteus.

Sequences	Proteus FDB	Proteus NEA	Rosetta
NHREF	24	20	35
Syntenin	31	33	38
DLG2	33	31	40

Table 4.14 – Pourcentage d'identité moyen à la séquence native

4.8.5 Logos des séquences obtenues

Finalement, nous montrons ici les séquences obtenues. Elle sont représentées sous forme de logos à la figure 4.18 pour les positions du cœur hydrophobe, et la figure ?? pour les positions exposées. L'accord avec les séquences naturelles sur les positions du cœur est très bon et l'accord sur les positions exposées est nettement moins bon.Mais au regard

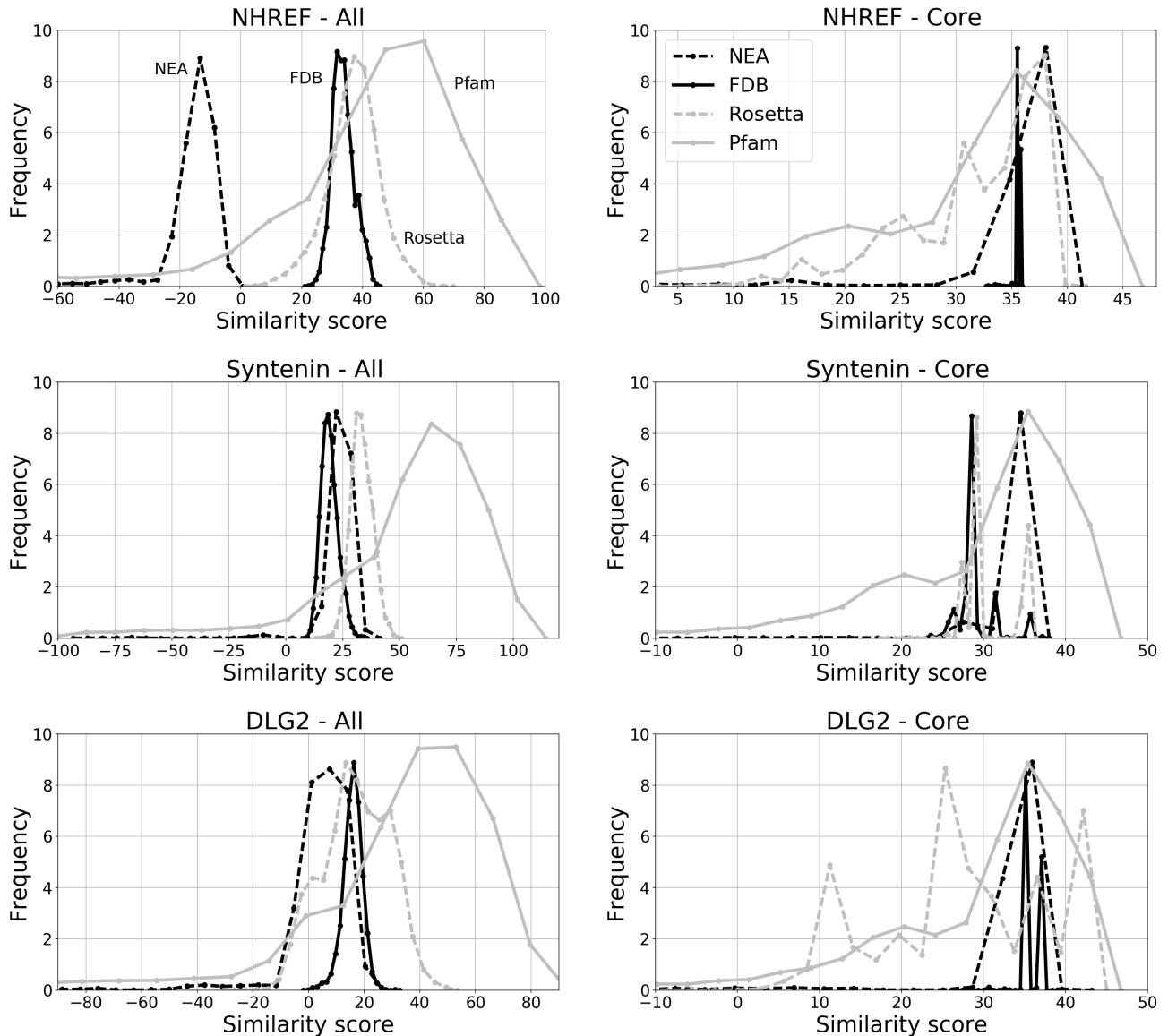


Figure 4.16 – Similarité des séquences Proteus (NEA et FDB), Rosetta , et des séquences de l’alignement Pfam RP55, sur toutes les positions à gauche et sur les positions du cœur à droite.

de la diversité des types aux positions exposées dans les séquences naturelles de Pfam le consensus entre les séquences naturelles est lui-même quasi inexistant.

4.9 Application : Croissance du noyau hydrophobe

Comme application de nos modèles optimisés, nous examinons la possibilité de concevoir du cœur hydrophobe des domaines PDZ. Chaque domaine PDZ est soumis à une simulation

4.9. Application : Croissance du noyau hydrophobe

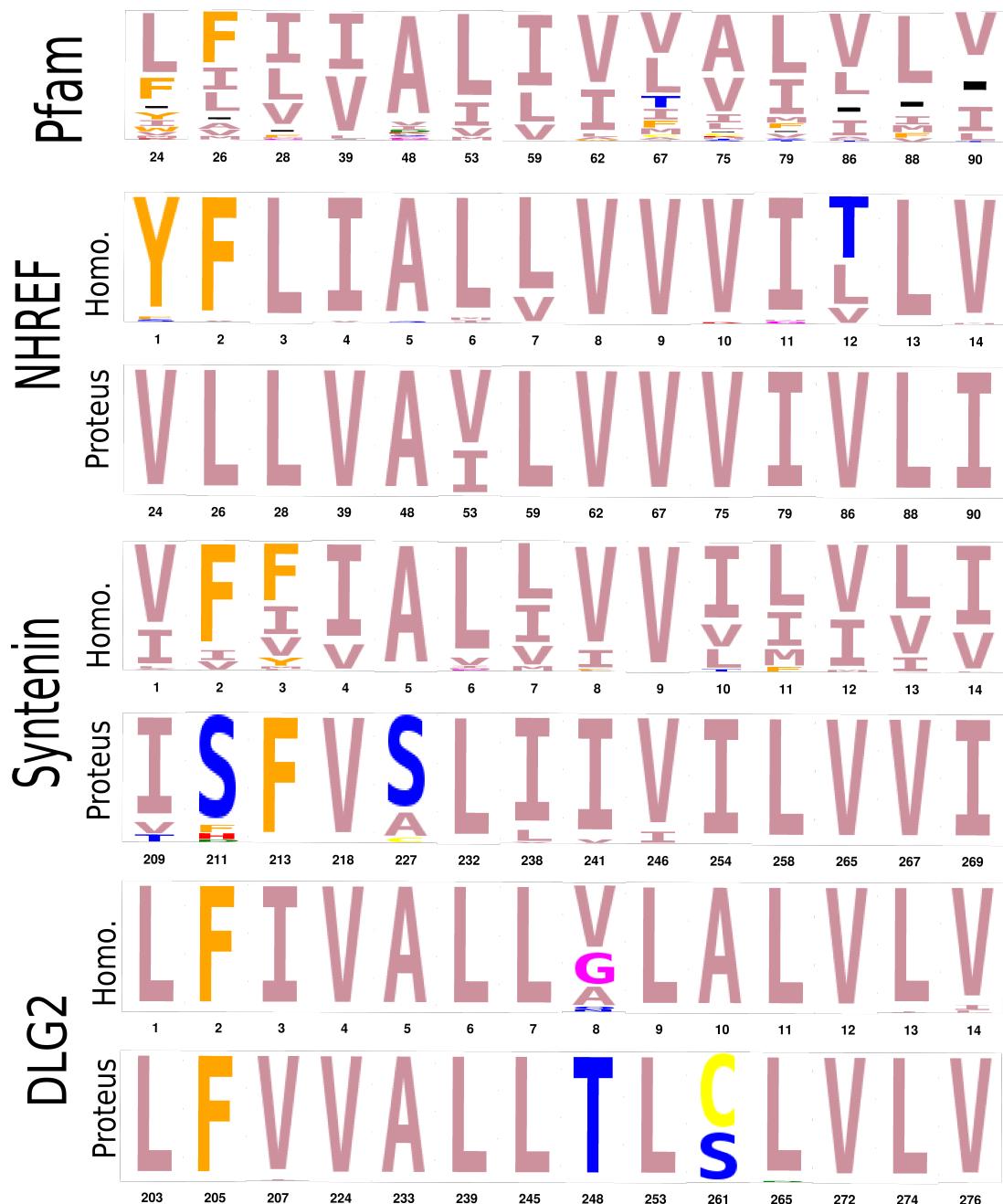


Figure 4.17 – Les séquences conçues par Proteus et les séquences naturelles représentées sous forme de logo pour les positions du cœur hydrophobe

REMC avec une succession de fonctions énergétiques biaisées qui favorisent de plus en plus les résidus hydrophobes. La première simulation comprend un terme d'énergie de biais $\delta = 0,4 \text{ kcal/mol}$ (par position) qui pénalise les types d'acides aminés hydrophobes (I,L,M,V,A,W,F et Y). Le biais augmente alors progressivement, est passé par les valeurs intermédiaires $\delta = 0,2 \text{ kcal/mol}$, $\delta = 0 \text{ kcal/mol}$ et $\delta = -0,2 \text{ kcal/mol}$. La dernière simulation

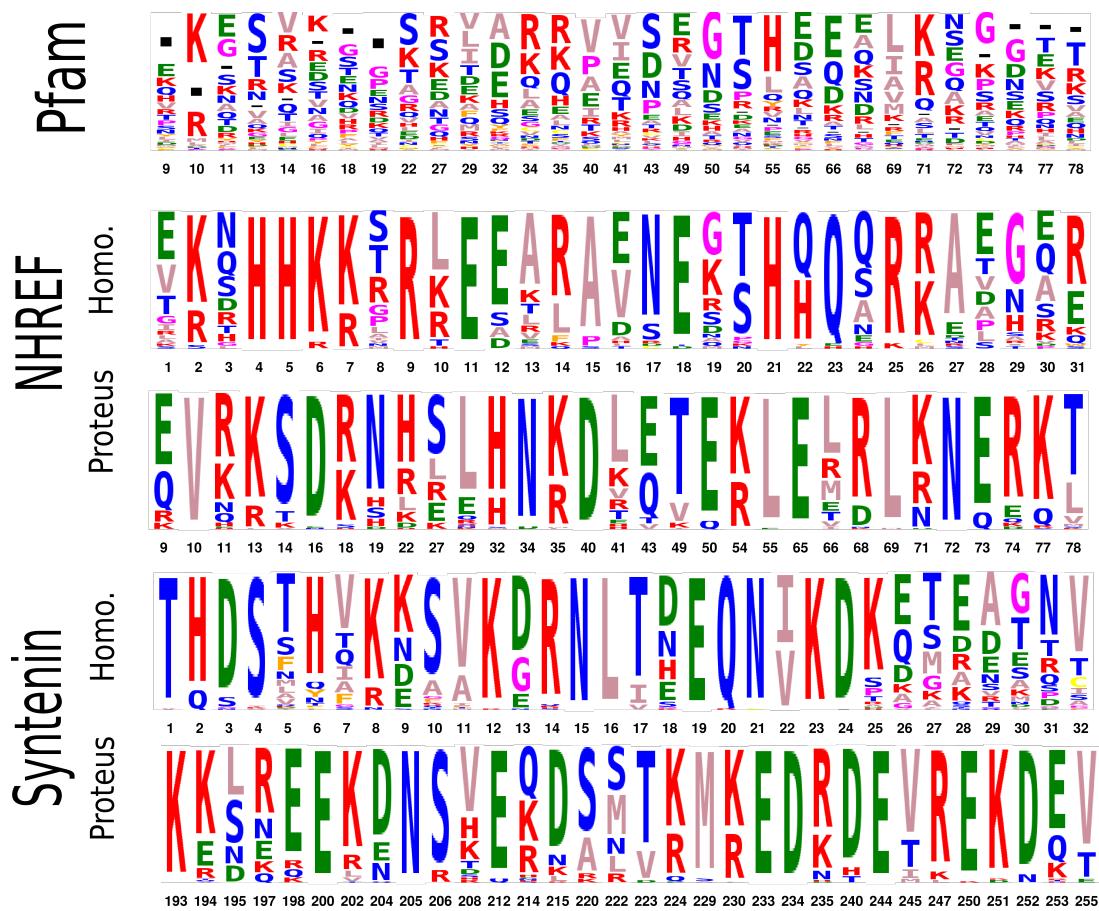
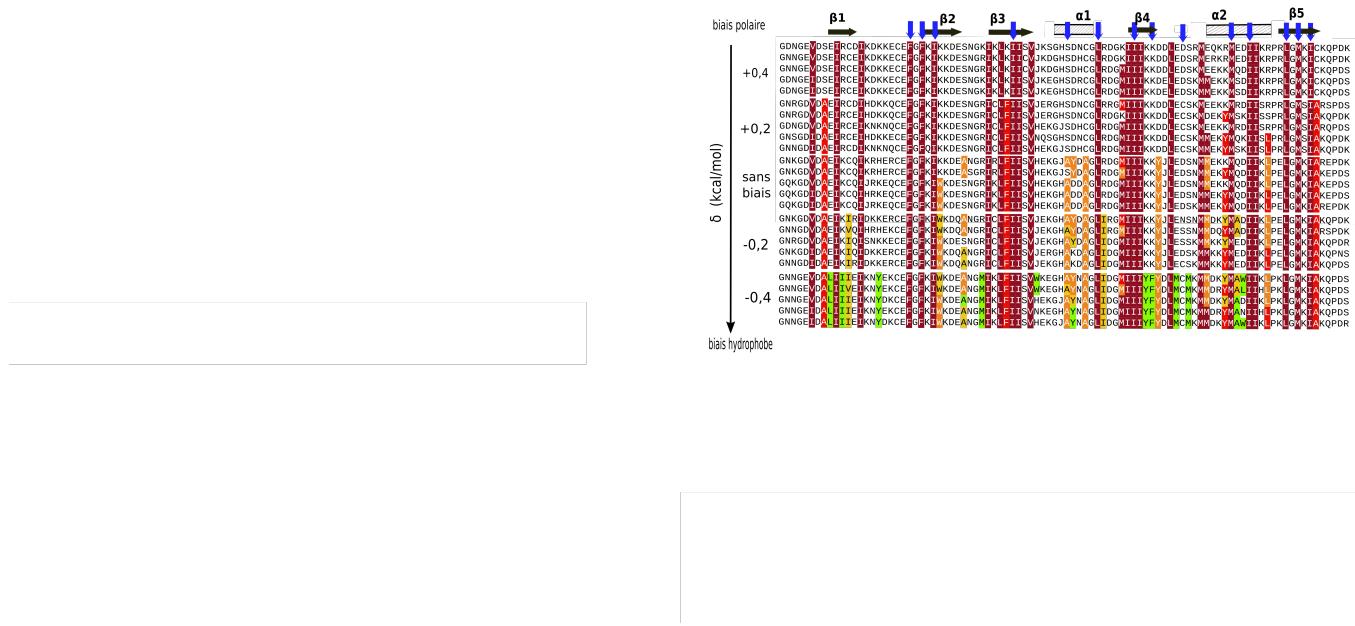


Figure 4.18 – Les séquences conçues par Proteus et les séquences naturelles représentées sous forme de logo pour les positions du cœur hydrophobe

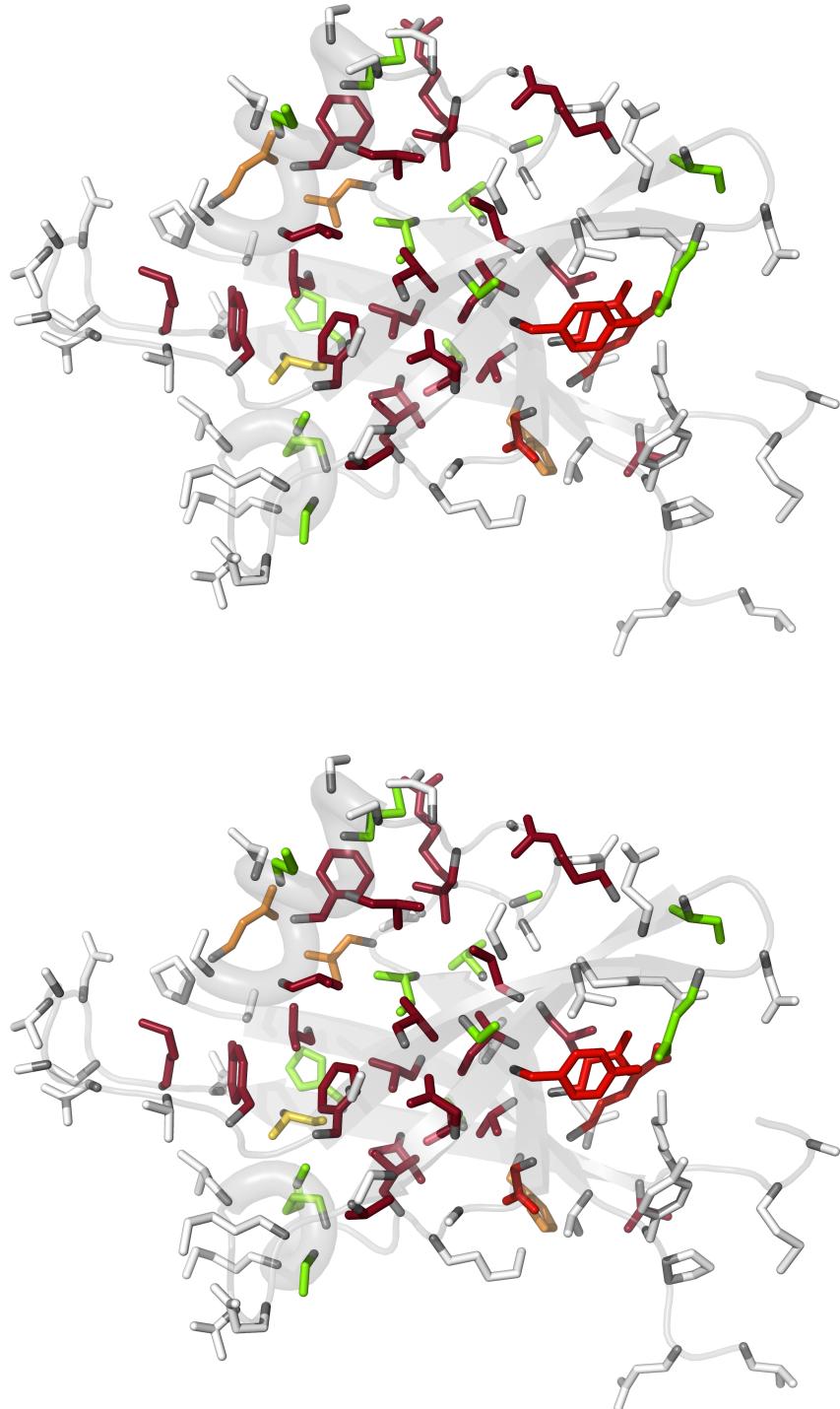
comprend un terme d'énergie de biais $\delta = -0.4 \text{ kcal/mol}$ (par position) qui favorise les types hydrophobes. En diminuant progressivement la valeur du biais d'énergie δ , nous « titrons » efficacement les résidus hydrophobes.

4.9. Application : Croissance du noyau hydrophobe



Les flèches bleues indiquent les positions du cœur hydrophobe PDZ défini à partir de notre sélection de 6 domaines. Chaque groupe de séquences est une sélection à δ fixé parmi les séquences de plus faible énergie.

Figure 4.19 – Structure native Tiam1 avec les hydrophobes pour des δ de -0,4,-0,2,0,0,2 et 0,4 sont représentés par un jeu de couleurs allant du rouge foncé au vert clair, en passant par le jaune.



Les résultats pour Tiam1 sont présentés à la figure ?? et à la figure 4.19. À la plus grande valeur de δ le cœur hydrophobe de Tiam1 est réduit à environ 10 positions (environ parce qu'il s'agit d'une sélection de séquences) d'acides aminés (sur 94) qui changent en un type polaire par rapport aux séquences sans biais. Les positions modifiées se situent principalement sur le bord extérieur du noyau. A la valeur intermédiaire de δ à $0,2\text{kcal/mol}$, le noyau hydrophobe ne compte plus que 4 ou 5 changements en type polaire. À la valeur de δ la plus négative, le cœur hydrophobe devient plus grand, s'étendant vers les régions de surface , avec globalement 14 positions polaires changées en type hydrophobe. Ainsi le nombre de positions modifiées est approximativement symétrique (environ +/- 12 changements), reflétant le biais. Environ 2 tiers des changements se produisent dans des éléments de structure secondaire. Dans l'ensemble, les propensions observées de chaque position à devenir polaire ou hydrophobe en présence d'un biais de pénalité petit ou grand d'énergie δ peuvent être considérées comme un indice de conception hydrophobe. Ici, 11 des 14 positions du cœur PDZ (toute sauf les positions 884 898 et 903) sont restées hydrophobes au plus haut niveau de biais polaire, avec à peu près 13 autres positions, indiquant que ces positions ont la plus grande propension à être hydrophobe. De plus, près de 14 positions ont basculé de polaire à hydrophobe au niveau de polarisation le plus élevé, indiquant que ces positions aussi ont une certaine propension a être hydrophobe. Les résultats pour Cask sont similaires, avec 11 positions changés en polaire au plus haut biais polaire et 9 changés en hydrophobe au plus haut biais hydrophobe, voir 4.21.

Nous introduisons également un indice pour décrire le nombre de changements relatifs de type d'acide aminé par unité d'énergie du biais. Cet indice ψ_h est défini comme le nombre δN de positions qui ont changées de non polaire à polaire, divisé par le produit de la variation δE dans l'énergie de polarisation et le nombre moyen N de position non polaire à biais nul. Nous appelons ψ_h la sensibilité hydrophobe. Pour le domaine PDZ Tiam1, ce calcul donne : $\psi_h = \frac{1}{N} \frac{\delta N}{\delta E} = 0,9$ changements par position et par kcal/mol. Pour Cask, la sensibilité hydrophobe est $\psi_h = 0,7$ changements par position et par kcal/mol.

4.10 Conclusion

4.10.1 modèle mis en œuvre

Nous avons paramétré notre modèle CPD pour la conception de domaine PDZ, mis en œuvre dans le logiciel Proteus. Pour la modélisation de l'état replié, nous avons utilisé un champ de force protéique de qualité. Nous avons effectué un premier paramétrage sur un ensemble de 8 domaines PDZ, dont 2 pour évaluer la transférabilité du paramétrage,

Chapitre 4. PDZ

Figure 4.20 – Structure native Cask avec les hydrophobes pour des δ de -0,4,-0,2,0,0,2 et 0,4 sont représentés par un dégradé allant du rouge foncé au vert clair, en passant par le jaune.

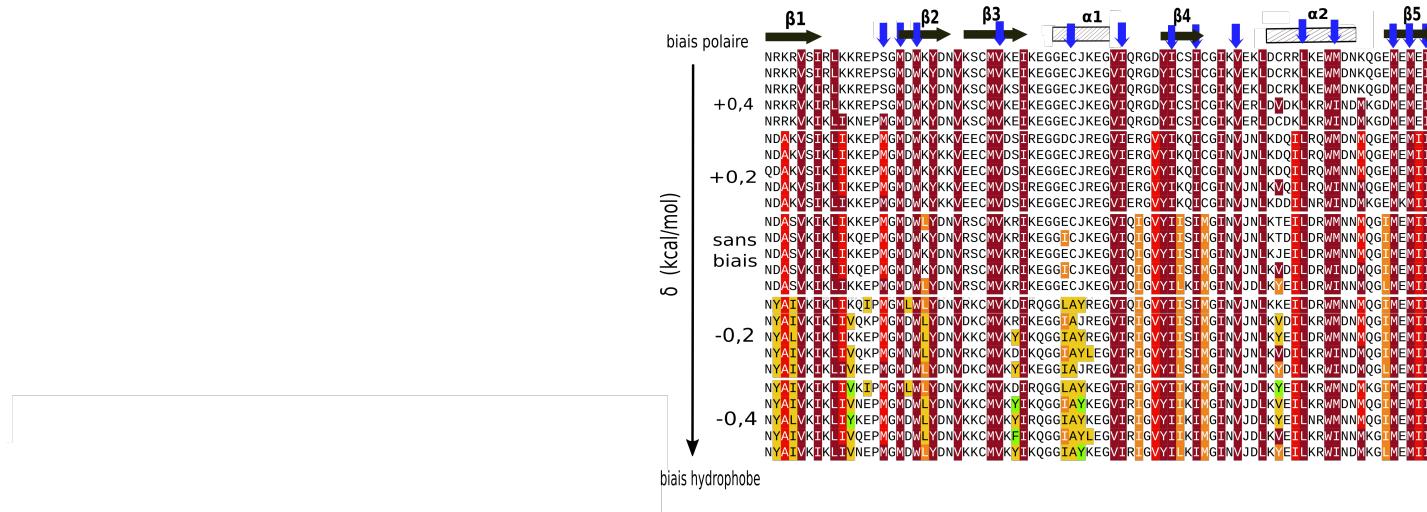


Figure 4.21 – Séquences Cask obtenues avec un delta des énergies de références à -0,4,-0,2,0,0,2 et 0,4 et la structure native. Les hydrophobes pour des deltas de -0,4,-0,2,0,0,2 et 0,4 sont représentés par un jeu de couleurs allant du rouge foncé au vert clair, en passant par le jaune.

avec un premier modèle de solvant « GB », le modèle NEA et une constante diélectrique ϵ_P égale à 8. Puis nous avons effectué de nouveaux paramétrages sur un ensemble de 3 domaines PDZ, avec une constante diélectrique ϵ_P égale à 4 et avec deux modèles de solvant, le NEA et un second modèle « GB », le modèle FDB. Pour les chaînes latérales, nous utilisons une bibliothèque de rotamères simple et discrète et une petite minimisation de chaque paire pendant le calcul de la matrice d'énergie pour atténuer l'approximation de la discréétisation des rotamères. La fonction d'énergie et la description des rotamères ont été testées de manière approfondie et ont démontré de très bonnes performances pour les tests de reconstruction de chaînes latérales ? (comparable au programme très populaire Scwrl4 (?)).

La représentation de l'état déplié utilise un modèle simple caractérisé par un ensemble de potentiels chimiques d'acide aminé empiriques ou énergies de référence. Ces énergies sont déterminées par une procédure de maximisation de vraisemblance, décrite ici, afin de reproduire la composition d'acides aminés d'homologues naturels soigneusement sélectionnés. L'état déplié utilisé ici bénéficie d'un raffinement supplémentaire, puisque des valeurs d'énergies de référence distinctes sont utilisées pour les positions d'acides aminés qui sont enfouis ou exposés à l'état replié.

Cette méthode suppose qu'il existe une structure résiduelle à l'état déplié, où certaines positions sont plus enfouies que d'autres. En outre, cela doit rendre le paramétrage plus robuste et moins sensible à la taille et à la structure des homologues naturels utilisés pour définir les compositions d'acide aminé cibles, car les fréquences d'acide aminé des positions exposées et des régions enfouies sont calculées séparément. En principe, cela double le nombre d'énergie de référence à ajuster. Cependant, nous avons réduit ce nombre en introduisant des classes de similarité d'acide aminé, avec une énergie de référence ajustable par classe. Cette contrainte est levée dans la seconde moitié des cycles d'optimisation. Lors de l'optimisation des énergies de référence, nous effectuons, des calculs de séquences pour chaque protéine de notre jeu de test où une position sur deux peut muter, soit la moitié des positions (à l'exception de Gly et Pro), avec une simulation distincte pour chaque moitié. De cette façon, lors de l'optimisation des paramètres, une position mutable est toujours entourée d'un environnement identique au type sauvage au moins sur les deux positions immédiatement voisines sur le squelette. Les calculs de conception des séquences s'appuient sur une méthode d'exploration Monte-Carlo avec échange de réplique, qui utilise plus d'un demi-milliard de pas par simulation et par réplique, et produit des milliers de séquences par simulation. Le modèle présente plusieurs limitations, dont la plupart se trouvent dans les implémentations et les applications du CPD. La première est l'utilisation de la stabilité des protéines comme seul critère de conception, sans prendre

en compte explicitement la spécificité du pli ??, la protection contre l'agrégation , ou des considérations fonctionnelles comme la liaison des ligands. Toutefois, nous notons que les tests superfamily n'ont pas entraînés de mauvaises affectations (séquences perçues comme préférant un autre pli SCOP), donc en pratique, la spécification du pli est réalisée. Une limitation supplémentaire est introduite par l'utilisation d'un squelette protéique fixe lors du calcul de la matrice d'énergie. En fait, le squelette n'est pas vraiment fixe. Certains mouvements sont autorisés, mais modélisés implicitement, à travers l'utilisation d'une constante diélectrique protéique supérieure à 1 ($\epsilon_p = 4$ ou 8) (?). Cette valeur diélectrique signifie que la structure protéique (y compris son squelette) est autorisée à se détendre ou à se réorganiser en réponse à une redistribution de charge associée à des mutations ou à un changement de rotamères de la chaîne latérale. Cependant, la réorganisation est modélisée non pas explicitement, mais implicitement (?), et elle n'implique pas de mouvement des centres atomiques ou de leur sphère de Van der Waals associée. Ainsi, le squelette ne peut ni se réorganiser en réponse à une répulsion stérique produite par des mutations ou des changements de rotamères ni se déplacer pour remplir l'espace laissé vide par une mutation. L'utilisation d'un squelette fixe peut être en partie compensée en concevant plusieurs structures PDZ. Par exemple, la mise en commun des séquences calculées sur 6 protéines a donné une entropie moyenne de séquence nettement plus proche de celle de l'ensemble expérimental Pfam.Une nouvelle méthode pour la conception de protéine multi backbone a récemment été développée dans Proteus, sur la base d'une méthode Monte-Carlo hybride qui préserve l'échantillonnage de la distribution de Boltzmann ?. Cette méthode pourra être appliquée dans les prochaines études. Une autre limitation de notre modèle est la nécessité, pour des résultats optimaux, de paramètres les énergies de référence spécifiquement pour un ensemble donné de protéines. Cette étape est bien automatisée et de façon très parallèle. Cependant, cela implique plusieurs choix qui sont partiellement arbitraires. Ceux-ci comprennent le choix d'un ensemble de domaines protéiques pour représenter la protéine ou la famille d'intérêt. Nous devons également choisir un seuil de similarité pour définir les homologues cibles à partir desquels sont calculées les compositions expérimentales d'acides aminés. Ici, nous avons choisi d'utiliser les homologues de chaque membre de la famille, de calculer leurs compositions, puis la moyenne sur l'ensemble des familles. Cette méthode a correctement fonctionné, mais d'autres choix sont possibles et des travaux complémentaires sont nécessaires pour pouvoir tirer des conclusions définitives sur ces choix.

Une autre limitation de notre modèle est l'approximation de la position des chaînes latérales en rotamères discrets. qui nécessite une certaine adaptation de la fonction d'énergie

pour éviter les affrontements stériques exagérés. La méthode utilisée ici est la méthode de minimisation de la paire de résidus décrite précédemment (?,?).

Une cinquième limitation est l'utilisation d'un modèle de solvatation additif par paire (comme dans la plupart des modèles CPD). Plus précisément, l'environnement diélectrique de chaque paire de résidus est supposé ici être celui de la structure native (ce qu'on appelle « Approvisionnement environnemental natif » ou NEA ?,?). Cela conduit à une fonction énergétique qui a la forme d'une somme sur les paires de résidus et peut être précalculée et stockée dans une matrice énergétique, qui sert alors de table de consultations pendant les simulations Monte-Carlo. Malgré cette approximation, le modèle a donné de bons résultats pour un grand nombre d'indices acide/base de référence, un problème très sensible au traitement électrostatique ?. Certaines de ces limitations sont supprimées dans le modèle FDB. En particulier, comme la fonction d'énergie est principalement basée sur la physique, elle a pu être améliorée par implémentations d'un calcul GB plus exact. Par ailleurs, il a été mis en place un modèle amélioré pour la solvatation hydrophobe (80), qui est plus rapide et plus précise que notre terme d'énergie de surface actuelle (article en préparation).

4.10.2 tests et application

Les séquences conçues par Proteus sont comparées aux séquences naturelles, à travers des tests de reconnaissance du pli, des calculs de similarité, des calculs d'entropie et des taux d'identité de séquences. Dans les simulations, nous concevons la totalité de la séquence de la protéine, de sorte que toutes les positions (à l'exception de Gly et Pro) peuvent muter librement, soumis à la seulement contrainte de générer une composition moyenne en acides aminés similaire à la composition moyenne expérimentale (à travers les énergies de références). Malgré la quasi-absence de contrainte expérimentale, les séquences obtenues ont une forte similitude globale avec les séquences naturelles de Pfam, mesurée par les scores de similarité Blosum40. Les scores obtenus sont, pour l'essentiel, comparables aux scores de similarité entre les paires de séquences Pfam entre elles. La similitude est très forte pour les résidus au cœur de la protéine, comme cela a été observé dans des études CPD précédentes (?,?). En revanche, pour les résidus pris sur l'ensemble de la protéine, les scores de similarité sont plus faibles, mais l'approximation FDB couplée à une constante diélectrique ϵ_p égale à 4 donne toujours des séquences similaires à des homologues naturels modérément éloignés. Notez que de nombreux résidus de surface sont impliqués dans des interactions fonctionnelles, comme les onze résidus de liaison aux peptides dans les domaines PDZ. Les résidus de surface sont également sélectionnés selon l'évolution pour éviter l'agrégation ou des adhésions indésirables. Ces contraintes fonctionnelles ne sont

Chapitre 4. PDZ

pas explicitement prises en compte dans notre protocole de design. Malgré ces difficultés sur les résidus de surface, la reconnaissance de pli avec l'outil Superfamily appliquée aux meilleurs modèles conçus est presque parfaite. Les tests de reconnaissance de plis antérieurs qui utilisaient une fonction d'énergie plus simple donnaient un taux de reconnaissance de pli inférieur, à environ 85% (pour un ensemble de tests plus large et plus diversifié) et des similitudes inférieures (?). De toute évidence, l'utilisation combinée d'un champ de force protéique amélioré, du solvant GB FDB et des énergies de référence spécifiques à la famille conduisent à des séquences calculées proche des séquences natives et sans doute meilleures. Les séquences Proteus ont également été comparées aux séquences obtenues avec le logiciel Rosetta , qui a lui-même été testé de manière approfondie. Sur la Base des scores de similarité de Blosum (par rapport aux séquences naturelles dans Pfam) et des tests de reconnaissance du pli, les séquences Proteus et Rosetta sont de même qualité. Cependant, Rosetta fait moins de mutations que Proteus ; de sorte que les scores d'identité, par rapport à la protéine de type sauvage correspondante, sont entre 7% et 9% plus haut chez Rosetta que pour la version FDB de notre modèle. Ce qui veut dire que Proteus modifie environ 5 positions en plus, en moyenne, par domaine PDZ.

Conclusion

XXX

Annexe 1 :Liste des positions actives pour chaque test

Conclusion

Nom	S_{Vois}	positions actives
1A81 1	10	10 13 16 84 86
1A81 2	10	20 21 24 27 116
1A81 3	10	35 38 56 105 107
1A81 4	10	44 47 52 65 67
1A81 5	10	82 84 86 87 90
1ABO 1	10	64 66 90 93 100
1ABO 2	10	72 74 80 104 111
1ABO 3	10	79 82 102 111 115
1ABO 4	10	83 86 104 105 106
1ABO 5	10	93 100 102 113 116
1BM2 1	10	101 106 140 141 146
1BM2 2	10	120 128 131 132 135
1BM2 3	10	58 61 127 128 129
1BM2 4	10	74 75 98 100 105
1BM2 5	10	85 87 95 110 128
1CKA 1	10	136 138 158 175 190
1CKA 2	10	149 166 169 171 181
1CKA 3	10	151 153 157 159 172
1CKA 4	10	164 170 172 184 187
1CKA 5	10	172 174 182 186 187
1G9O 1	10	10 13 54 57 92
1G9O 2	10	15 39 42 54 57
1G9O 3	10	24 26 28 39 42
1G9O 4	10	48 53 57 59 88
1G9O 5	10	75 78 79 86 88
1M61 1	10	12 20 23 24 27
1M61 2	10	17 20 24 37 49
1M61 3	10	27 33 51 100 102
1M61 4	10	5 8 10 11 36
1M61 5	10	59 71 84 87 94
1O4C 1	10	20 21 32 34 46
1O4C 2	10	2 71 79 81 82
1O4C 3	10	33 45 63 71 73
1O4C 4	10	43 45 63 71 85
1O4C 5	10	8 33 82 83 86
1R6J 1	10	194 237 239 270 272
1R6J 2	10	199 201 211 218 232
1R6J 3	10	213 218 227 232 238
1R6J 4	10	221 227 232 267 269
1R6J 5	10	241 254 258 267 269
2BYG 1	10	189 191 221 244 246
2BYG 2	10	205 224 239 245 248
2BYG 3	10	232 233 265 272 274
2BYG 4	10	238 240 243 276 278
2BYG 5	10	253 261 264 265 274

Table 15 – Les tests avec cinq positions actives

Nom	S_{Vois}	positions actives
1A81 1	10	13 15 39 41 53 86 89 90 93 103
1A81 2	10	39 41 53 55 64 66 76 89 92 103
1A81 3	10	51 53 64 66 68 74 76 82 88 92
1A81 4	10	76 82 87 88 90 91 92 95 97 99
1A81 5	10	9 10 11 16 41 51 53 66 88 89
1ABO 1	10	64 72 74 79 89 91 101 103 108 111
1ABO 2	10	66 68 80 82 88 90 100 102 104 111
1ABO 3	10	69 70 72 74 80 81 106 113 114 115
1ABO 4	10	71 78 83 84 94 99 101 104 105 106
1ABO 5	10	72 79 82 94 99 102 104 106 111 115
1BM2 1	10	119 120 121 122 123 125 131 134 135 140
1BM2 2	10	125 126 127 129 130 133 134 136 137 147
1BM2 3	10	83 99 101 106 108 135 140 141 146 148
1BM2 4	10	85 95 97 110 118 120 125 128 131 132
1BM2 5	10	99 101 106 139 140 141 142 143 144 146
1CKA 1	10	134 135 160 161 162 173 174 175 176 179
1CKA 2	10	137 139 143 151 153 157 159 172 182 186
1CKA 3	10	138 140 147 149 150 155 166 169 181 188
1CKA 4	10	140 141 153 154 155 157 174 175 184 186
1CKA 5	10	151 153 157 166 168 173 174 176 178 179
1G9O 1	10	10 11 13 14 15 16 53 54 57 92
1G9O 2	10	15 17 24 26 39 42 48 51 53 88
1G9O 3	10	26 28 39 42 48 53 57 59 88 90
1G9O 4	10	34 35 58 60 68 70 74 75 89 91
1G9O 5	10	71 73 74 77 80 81 82 83 84 85
1M61 1	10	10 12 20 23 24 27 35 49 102 104
1M61 2	10	17 20 21 24 37 39 40 47 49 58
1M61 3	10	34 36 46 48 59 61 71 83 84 87
1M61 4	10	5 6 11 36 46 48 61 69 83 84
1M61 5	10	59 61 70 71 75 77 83 86 87 92
1O4C 1	10	31 33 45 47 61 63 73 86 89 100
1O4C 2	10	50 51 52 53 63 72 73 77 85 89
1O4C 3	10	61 62 63 71 72 73 79 85 88 89
1O4C 4	10	73 74 75 76 77 89 92 94 96 101
1O4C 5	10	90 91 93 96 98 99 101 102 103 104
1R6J 1	10	193 194 195 197 199 218 232 236 267 269
1R6J 2	10	199 209 211 213 218 227 232 238 265 267
1R6J 3	10	201 204 205 209 211 218 241 258 265 267
1R6J 4	10	209 211 213 218 227 238 241 258 265 267
1R6J 5	10	238 240 241 242 246 257 258 261 265 267
2BYG 1	10	194 196 203 205 224 233 239 245 274 276
2BYG 2	10	203 205 207 224 227 233 239 243 245 276
2BYG 3	10	206 207 222 245 248 251 253 256 261 264 265
2BYG 4	10	221 222 245 248 251 253 256 261 264 265
2BYG 5	10	247 248 249 250 251 252 259 262 263 275

Table 16 – Les tests avec dix positions actives

Conclusion

Nom	S_{Vois}	positions actives
1A81 1	1	9 11 12 13 15 16 17 19 20 25 28 39 40 41 42 43 44 45 114 117
1A81 2	1	9 11 12 13 15 16 17 19 20 25 28 39 40 41 42 43 44 45 47 117
1A81 3	1	9 11 12 13 15 16 17 19 19 41 43 48 51 68 74 84 86 109 114 117
1A81 4	1	12 13 15 16 17 19 20 25 28 39 40 41 42 43 44 45 47 86 114 117
1A81 5	1	13 15 16 19 41 43 48 51 60 64 68 70 71 74 84 86 87 88 109 114 117
1ABO 1	1	64 66 67 68 82 86 87 88 89 90 91 101 102 102 103 103 108 111 113 116
1ABO 2	1	64 65 65 66 67 84 87 88 89 90 91 93 100 101 102 103 108 111 113 116
1ABO 3	1	65 66 67 87 88 89 90 91 93 94 95 100 101 102 103 106 108 111 113 116
1ABO 4	1	64 65 66 67 69 87 88 89 90 91 93 100 101 102 103 106 108 111 113 116
1ABO 5	1	66 67 68 82 86 87 88 89 90 91 93 100 101 102 103 106 108 111 113 116
1BM2 1	1	55 56 60 61 62 83 84 85 86 87 95 97 99 110 118 125 127 133 150 152
1BM2 2	1	55 56 60 61 62 83 84 85 86 87 95 97 99 110 118 125 127 128 129 152
1BM2 3	1	55 56 58 60 61 62 64 67 69 73 83 84 85 86 87 129 132 133 150 152
1BM2 4	1	55 56 60 61 62 69 83 84 85 86 87 95 97 99 110 129 132 133 150 152
1BM2 5	1	58 60 60 61 61 62 64 67 69 73 75 83 84 85 86 129 132 133 150 152
1CKA 1	1	134 135 136 137 138 139 150 151 160 161 162 163 164 170 171 172 173 179 189 190
1CKA 2	1	134 135 136 137 139 150 151 153 160 161 162 163 164 170 171 172 173 179 189 190
1CKA 3	1	134 136 137 139 150 151 157 158 160 161 162 163 164 170 171 172 173 179 189 190
1CKA 4	1	136 137 139 150 151 153 158 159 160 161 162 163 164 170 171 172 173 179 189 190
1CKA 5	1	137 139 150 151 153 158 160 161 162 163 164 170 171 172 173 174 175 179 189 190
1G90 1	1	9 10 11 13 14 15 31 34 38 54 57 58 60 68 90 91 92 94 95 96
1G90 2	1	9 11 13 14 15 16 31 34 38 54 57 58 60 68 90 91 92 94 95 96
1G90 3	1	9 11 13 14 15 31 34 38 54 55 57 58 60 68 90 91 92 94 95 96
1G90 4	1	9 11 13 15 16 17 54 57 58 59 60 61 68 89 90 91 92 94 95 96
1G90 5	1	10 11 13 15 16 17 54 57 58 60 61 68 89 90 90 91 92 94 95 96
1M61 1	1	34 35 36 37 38 39 42 46 47 48 49 50 61 63 69 71 77 78 81 82
1M61 2	1	35 36 37 38 39 42 46 47 48 49 50 61 63 69 71 77 78 81 82 83
1M61 3	1	38 39 42 46 47 48 49 50 61 63 69 71 77 78 81 82 83 84 85 87
1M61 4	1	42 46 47 48 49 50 61 63 69 71 77 78 81 82 83 84 85 87 88 98
1M61 5	1	5 7 50 61 63 69 71 77 78 81 82 83 84 85 87 88 98 103 104 109
1O4C 1	1	32 33 34 35 43 45 63 65 71 73 79 80 81 82 83 84 85 86 87 89
1O4C 2	1	3 4 5 7 8 9 11 17 31 32 33 34 35 43 45 63 65 71 73 79
1O4C 3	1	1 3 4 5 6 7 8 9 11 17 31 32 33 35 43 45 65 81 82 83
1O4C 4	1	1 2 3 4 5 6 7 8 9 11 12 13 14 17 19 35 65 81 82 83
1O4C 5	1	1 3 4 5 6 7 8 9 11 12 17 31 32 33 34 35 65 81 82 83
1R6J 1	1	193 194 195 197 214 215 217 218 233 235 236 237 239 240 241 242 247 269 270 273
1R6J 2	1	193 194 197 198 199 217 233 235 236 237 238 239 240 241 242 247 268 270 272 273
1R6J 3	1	193 195 197 217 233 235 236 239 240 241 242 244 245 247 268 269 270 270 272 273
1R6J 4	1	193 195 197 217 233 235 236 237 239 241 242 244 245 247 268 269 270 272 273 273
1R6J 5	1	193 194 197 198 199 233 236 237 239 240 241 247 268 268 269 270 270 272 273
2BYG 1	1	186 187 188 189 190 191 192 215 216 219 244 246 270 271 273 274 278 280 281 282
2BYG 2	1	186 187 188 189 190 215 216 219 221 223 240 243 270 271 273 274 278 280 281 282
2BYG 3	1	186 187 188 189 190 215 216 219 221 223 240 243 244 270 271 273 278 280 281 282
2BYG 4	1	186 187 188 189 190 215 216 219 221 223 240 243 244 270 274 276 278 280 281 282
2BYG 5	1	187 189 190 191 192 215 216 219 221 223 240 243 244 270 274 276 278 280 281 282

Table 17 – Les tests avec vingt positions actives

Nom	S_{Vois}	positions actives
1A81 1	1	9 11 12 13 15 16 17 19 20 25 26 27 28 29 36 38 39 40 41 42 43 48 51 68 74 84 86 109 114 117
1A81 2	1	9 10 11 12 13 15 16 17 19 20 25 28 39 41 43 48 51 68 74 83 84 86 87 88 90 91 93 109 114 117
1A81 3	1	9 11 12 13 15 16 17 19 20 25 27 28 36 38 39 40 41 42 43 48 51 68 74 84 86 109 114 117
1A81 4	1	9 10 11 12 13 15 16 17 19 20 25 28 36 39 40 41 42 43 44 45 48 51 68 74 84 86 109 114 117
1A81 5	1	9 10 11 12 13 15 16 17 19 20 25 28 39 40 41 42 43 44 45 47 48 51 52 68 74 84 86 109 114 117
1ABO 1	1	64 65 66 67 68 70 71 72 75 78 79 80 81 82 83 86 87 88 89 90 91 93 100 101 102 103 108 111 113 116
1ABO 2	1	64 65 66 67 68 72 75 78 80 81 82 83 84 86 87 88 89 90 91 93 94 100 101 102 103 104 108 111 113 116
1ABO 3	1	64 66 67 68 70 71 72 78 82 86 87 88 89 90 91 93 94 95 96 99 100 101 102 103 104 105 108 111 113 116
1ABO 4	1	64 65 66 67 70 71 72 68 82 86 87 88 89 90 91 93 94 95 96 99 100 101 102 103 104 105 108 111 113 116
1ABO 5	1	65 66 67 70 71 72 75 78 80 82 86 87 88 89 90 91 93 94 95 96 99 100 101 102 103 108 111 113 116
1BM2 1	1	55 56 58 60 61 62 83 84 85 86 87 95 97 99 110 118 119 120 121 122 125 127 128 129 130 131 132 133 150 152
1BM2 2	1	56 58 60 61 62 83 84 85 86 87 95 97 99 110 118 119 120 121 122 123 125 127 128 129 130 131 132 133 150 152
1BM2 3	1	58 60 61 62 83 84 85 86 87 95 97 99 108 110 118 120 121 122 123 125 127 128 129 132 133 134 135 150 152
1BM2 4	1	55 56 58 60 61 62 83 84 85 86 87 95 97 99 108 109 110 118 120 121 125 127 128 129 130 131 132 133 150 152
1BM2 5	1	56 58 60 61 62 67 83 84 85 86 87 95 97 99 110 111 112 113 115 118 125 127 128 129 130 131 132 133 150 152
1CKA 1	1	134 135 136 137 139 140 141 142 143 144 146 147 148 149 150 151 158 159 160 161 162 163 164 170 171 172 173 179 189 190
1CKA 2	1	134 135 136 137 139 143 144 146 147 148 149 150 151 158 159 160 161 162 163 164 170 171 172 173 179 186 187 188 189 190
1CKA 3	1	135 136 137 139 144 146 147 148 149 150 151 157 158 159 160 161 162 163 163 164 170 171 172 173 179 186 187 188 189 190
1CKA 4	1	136 137 139 140 141 142 143 144 146 147 148 151 158 159 160 161 162 163 164 170 171 172 173 179 184 186 187 188 189 190
1CKA 5	1	134 136 137 139 140 141 142 143 144 146 147 148 151 158 159 160 161 162 163 164 170 171 172 173 179 182 187 188 189 190
1G9O 1	1	9 10 11 13 15 24 31 34 38 40 41 42 43 46 48 49 50 51 54 57 58 60 68 89 90 91 92 94 95 96
1G9O 2	1	9 11 13 15 31 32 34 38 40 41 42 43 46 48 49 50 51 54 57 58 60 68 89 90 91 92 94 95 96
1G9O 3	1	9 10 11 13 15 31 32 34 38 40 41 42 43 46 48 49 50 51 54 57 58 60 68 89 90 91 92 94 95 96
1G9O 4	1	10 11 13 14 15 31 32 34 38 40 41 42 43 46 48 49 50 51 54 57 58 60 61 68 89 90 91 92 94 95 96
1G9O 5	1	10 11 13 14 15 31 32 40 41 42 43 46 48 49 50 51 54 57 58 60 61 62 68 87 89 90 91 92 94 95 96
1M61 1	1	12 14 15 20 34 35 36 37 38 39 42 46 47 48 49 50 61 63 69 71 77 78 81 82 83 84 85 87 88 98
1M61 2	1	6 7 8 10 11 12 14 15 20 21 34 35 36 37 38 39 42 46 47 48 49 50 61 63 69 71 77 78 81 82
1M61 3	1	5 7 35 36 37 38 39 42 46 47 48 49 50 61 63 69 71 77 78 81 82 83 84 85 87 88 98 103 104 109
1M61 4	1	7 8 10 11 12 14 15 20 34 35 36 37 38 39 42 46 47 48 49 50 61 63 69 71 77 78 81 82 83 84
1M61 5	1	8 10 11 12 14 15 20 34 35 36 37 38 39 42 46 47 48 49 50 61 63 69 71 77 78 81 82 83 84 85
1O4C 1	1	1 2 3 4 5 6 7 8 9 11 17 31 32 33 34 35 43 45 63 65 71 81 73 79 80 81 82 83 90 91 92 93 96
1O4C 2	1	1 3 4 5 7 8 9 11 17 31 32 33 34 35 43 45 63 65 71 73 79 80 81 82 83 90 91 92 93 96
1O4C 3	1	1 3 4 5 6 7 8 9 11 17 31 32 33 34 35 43 45 63 65 71 73 79 80 81 82 83 90 91 92 93 96
1O4C 4	1	1 3 4 5 7 8 9 11 17 31 32 33 34 35 43 45 63 65 71 73 79 80 81 82 83 84 91 92 93 96
1O4C 5	1	1 3 4 5 6 7 8 9 11 17 31 32 33 34 35 43 45 63 65 71 73 79 80 81 82 83 84 92 93 96
1R6J 1	1	193 194 195 197 198 199 217 218 219 220 221 222 223 224 225 226 227 228 229 233 235 236 237 239 247 268 269 270 272 273
1R6J 2	1	193 194 195 197 198 199 217 220 221 222 223 224 225 226 227 228 229 233 235 236 237 239 240 247 268 269 270 272 273
1R6J 3	1	193 194 195 197 198 199 208 217 220 221 222 223 224 225 226 227 228 229 230 233 235 236 237 239 247 268 269 270 272 273
1R6J 4	1	193 194 195 197 198 199 217 220 221 222 223 224 225 226 227 228 229 233 235 236 237 239 240 247 249 268 269 270 272 273
1R6J 5	1	194 195 197 198 199 217 220 221 222 223 224 225 226 227 228 229 233 235 236 237 239 240 247 249 268 269 270 272 273
2BYG 1	1	186 187 188 189 190 191 192 215 216 219 221 223 240 244 246 250 251 252 253 254 255 256 257 259 260 278 280 281 282
2BYG 2	1	186 187 188 189 190 191 192 215 216 219 221 223 240 244 246 251 252 253 254 255 256 257 259 260 278 278 280 281 282
2BYG 3	1	186 187 188 189 190 191 192 215 216 219 221 223 240 244 246 251 252 253 254 255 256 257 259 260 278 246 280 281 282
2BYG 4	1	186 187 188 189 190 191 192 215 216 219 221 223 240 244 245 246 251 252 253 254 255 256 257 259 260 278 278 281 282
2BYG 5	1	186 187 188 189 190 191 192 215 216 219 221 223 240 244 245 246 251 252 253 254 255 256 257 259 260 278 246 278 280 281 282

Table 18 – Les tests avec trente positions actives

Résum

Titre de la thèse

XXX

Mots-clés : motclé1, motclé2, motclé3

Abstract

Thesis title

XXX

Keywords: keyword1, keyword2, keyword3