



---

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
<b>1 le « CPD » : Conception de protéine par ordinateur</b>	<b>5</b>
1.1 l'espace des séquences-conformations . . . . .	6
1.1.1 l'état replié . . . . .	7
les chaînes latérales . . . . .	7
Le squelette . . . . .	8
1.1.2 l'état deplié . . . . .	8
1.2 l'énergie de conformation . . . . .	9
1.2.1 La mécanique moléculaire . . . . .	9
1.2.2 Les interactions liées . . . . .	10
1.2.3 Les interactions non liées . . . . .	11
1.2.4 D'autres approches . . . . .	11
1.3 Modélisation du solvant . . . . .	11
1.3.1 modèle de solvant explicite . . . . .	12
1.3.2 modèle implicite . . . . .	13
1.3.3 modèle CASA . . . . .	14
1.3.4 modèle Poisson Boltzmann . . . . .	15
1.3.5 modèle Born Généralisé . . . . .	16
1.4 l'algorithme d'exploration . . . . .	17
1.4.1 Algorithme du champ moyen . . . . .	19
1.4.2 Dead-End Elimination . . . . .	19
1.4.3 le CFN . . . . .	21

## Table des matières

---

1.4.4	l'heuristique Multistart Steepest Descent (MSD) . . . . .	24
1.4.5	Algorithme génétique . . . . .	25
1.4.6	Monte-Carlo . . . . .	26
1.5	Les programmes CPD . . . . .	31
1.6	Les succès . . . . .	31
<b>2</b>	<b>Méthodes</b>	<b>33</b>
2.1	Proteus . . . . .	33
2.1.1	deroulement de Proteus . . . . .	33
2.1.2	décomposition en paire du terme de surface . . . . .	34
2.1.3	« Native Environnement Approximation » (NEA) . . . . .	34
2.1.4	« Fluctuating Dielectric Boundary » (FDB) . . . . .	35
2.1.5	La matrice d'énergie . . . . .	36
2.1.6	l'état déplié . . . . .	37
2.1.7	les groupes dans proteus . . . . .	37
2.1.8	modèle de données . . . . .	37
2.1.9	Les entrées_sorties . . . . .	37
	Le fichier de configuration . . . . .	37
	Les fichiers « backbone » et « parwise » . . . . .	37
	Les fichiers de sortie . . . . .	39
2.2	Description des tests de comparaisons d'algorithme . . . . .	39
2.2.1	Sélection des positions . . . . .	40
	Ensemble «Tout actif» . . . . .	42
	L'ensemble «nombre d'actifs limité» . . . . .	43
	le choix des positions actives . . . . .	44
	positions en interactions . . . . .	44
	choix des positions actives . . . . .	44
2.2.2	Définition de protocole comparable . . . . .	45
	Protocole heuristique . . . . .	46
	Protocoles Monte-Carlo . . . . .	46
	Protocoles "Replica Exchange" . . . . .	47
	Distribution des énergies en fonction des températures . . . . .	48
	Trajectoires dans l'espace des températures . . . . .	48
	Protocoles Toulbar2 . . . . .	49
2.2.3	Outils d'analyse des données . . . . .	50
	Superfamily/SCOP . . . . .	50

Taux d'identité de séquences . . . . .	51
Taux d'identité par position . . . . .	51
Alignements Pfam . . . . .	51
Score BLOSUM . . . . .	51
similarité d'un ensemble à une famille Pfam . . . . .	52
<b>3 Comparing stochastic search algorithms for computational protein design</b>	<b>53</b>
3.1 Introduction . . . . .	54
3.2 Methods . . . . .	56
3.2.1 Monte Carlo : general framework . . . . .	56
3.2.2 MC and REMC : implementation details . . . . .	58
3.2.3 Heuristic sequence optimization . . . . .	59
3.2.4 Cost function network method . . . . .	60
3.2.5 Energy function . . . . .	60
3.2.6 Test systems and preparation . . . . .	61
3.2.7 Sequence characterization . . . . .	61
3.3 Results . . . . .	62
3.3.1 Quality of the designed sequences . . . . .	62
3.3.2 Finding the GMEC . . . . .	65
CPU and memory limits for each method . . . . .	65
Optimal sequences/structures with up to 10 designed positions . . .	68
Optimal sequences with 20 or 30 designed positions . . . . .	71
3.3.3 Density of states above the GMEC . . . . .	72
3.4 Conclusions . . . . .	76
<b>4 PDZ</b>	<b>81</b>
4.1 Introduction . . . . .	81
4.2 Le modèle d'état déplié . . . . .	82
4.2.1 Les énergies de référence . . . . .	82
4.2.2 La vraisemblance des énergies de référence . . . . .	83
4.2.3 Recherche du maximum de vraisemblance . . . . .	84
4.3 Méthodes de calcul . . . . .	86
4.3.1 Fonction énergétique efficace pour l'état replié . . . . .	86
4.3.2 Les énergies de référence de l'état déplié . . . . .	87
4.4 Outils d'analyse de séquences . . . . .	89
4.4.1 Superfamily/SCOP . . . . .	89

## **Table des matières**

---

4.4.2	Taux d'identité de séquences . . . . .	89
4.4.3	Taux d'identité par position . . . . .	90
4.4.4	Alignements Pfam . . . . .	90
4.4.5	Score BLOSUM . . . . .	90
4.4.6	Similarité d'un ensemble à un alignement Pfam . . . . .	91
4.4.7	Entropie par position . . . . .	91
4.5	Séquences expérimentales et modèles structurels . . . . .	92
4.5.1	L'ensemble des protéines PDZ . . . . .	92
4.5.2	Alignements Blast croisés . . . . .	92
4.5.3	Sélection des homologues . . . . .	92
4.5.4	Alignements des protéines expérimentales et leurs homologues . . . . .	94
4.5.5	Similarité des homologues . . . . .	94
4.5.6	Les fréquences d'acides aminés . . . . .	96
4.6	Séquences calculées . . . . .	109
4.6.1	simulation Monte-Carlo . . . . .	109
4.6.2	Génération de séquence Rosetta . . . . .	110
4.6.3	Caractérisation des séquences calculées . . . . .	110
4.7	Résultats du modèle NEA . . . . .	111
4.7.1	optimisation du modèle de l'état déplié . . . . .	111
4.7.2	Tests de reconnaissance de famille . . . . .	113
4.7.3	Séquences et diversité de séquences . . . . .	113
4.7.4	Scores de similarité Blosum . . . . .	114
4.7.5	Tests de validation croisée . . . . .	115
4.8	Résultats du modèle FDB . . . . .	117
4.8.1	optimisation du modèle de l'état déplié . . . . .	117
4.8.2	Tests de reconnaissance de famille . . . . .	118
4.8.3	Scores de similarité Blosum . . . . .	118
4.8.4	Taux d'identité à la séquence native . . . . .	121
4.8.5	Logos des séquences obtenues . . . . .	121
4.9	Application : Croissance du noyau hydrophobe . . . . .	122
4.10	Conclusion . . . . .	148
4.10.1	modèle mis en œuvre . . . . .	148
4.10.2	tests et application . . . . .	150
<b>Conclusion</b>		<b>153</b>
<b>Bibliographie</b>		<b>161</b>

# Liste des figures

1.1	Le CPD dans proteus . . . . .	6
1.2	<b>organisation des dipôles des molécules d'eau autour d'un soluté sphérique chargé négativement à sa surface</b> . . . . .	15
1.3	Principe du l'algorithme du Depth-First Branch and Bound . . . . .	23
1.4	Exemple de transformation EPT . . . . .	24
1.5	L'algorithle génétique . . . . .	32
2.1	<b>La matrice d'énergies</b> Sur cet exemple, un polypeptide de 6 résidues, chaque position posséde type d'acide aminés possibles et 3 rotamères possibles (2 pour le type A et 1 pour le type B). La matrice organise toutes les interactions de paires de chaîen latérales possibles. Les interactions impliquant le résidu numéro 2 sont dans la bande jaune de la matrice, les interactions impliquant le résidu numéro 3 sont dans la bande bleue.Les points rouge et vert correspondent aux interactions noté par les flèches rouge et verte à gauche(La matrice est symétrique , ici les interactions sont repräsentées qu'une seule fois). . . . .	36
2.2	Les principales strucrures « physiques » dans proteus . . . . .	39
2.3	Les principales strucrures « physiques » dans proteus . . . . .	40
2.4	Les principales strucrures « logiques » dans proteus . . . . .	41
2.5	Les principales strucrures « dynamiques » dans proteus . . . . .	42
2.6	Les fichiers en entrée . . . . .	42
2.7	Les fichiers en sortie de proteus . . . . .	43
2.8	Distribution des énergies selon la température (1A81 protocole RE8b1). . .	49
2.9	Variation de la température pendant la trajectoire de chaque marcheur (cas exemple de protocoles). . . . .	50
3.1	width=1cm . . . . .	63
3.2	width=1cm . . . . .	65

## Liste des figures

---

3.3	width=1cm . . . . .	66
3.4	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. . . . .	68
3.5	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.	69
3.6	width=1cm . . . . .	75
4.1	le cœur PDZ sélectionné . . . . .	95
4.2	L’alignement de notre sélection de séquences homologues à la protéine NHREF (code PDB :1G9O) . . . . .	97
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 . . . . .	98
4.4	L’alignement de notre sélection de séquences homologues à la protéine INAD (code PDB :1IHJ ) . . . . .	99
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 . . . . .	100
4.6	L’alignement de notre sélection de séquences homologues à la protéine Syntenin (code PDB 1R6J) . . . . .	101
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 . . . . .	102
4.8	L’alignement de notre sélection de séquences homologues à la protéine GRIP (code PDB : 1N7E) . . . . .	103

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 . . . . .	104
4.10 L'alignement de notre sélection de séquences homologues à la protéine DLG2 (code PDB 2BYG) . . . . .	105
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 . . . . .	106
4.12 L'alignement de notre sélection de séquences homologues à la protéine PSD95 (code PDB 3K82) . . . . .	107
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 . . . . .	108
4.14 Similarité des séquences des 6 protéines produites par Proteus et Rosetta à l'alignement Pfam RP55, sur l'ensemble des positions. . . . .	116
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.	117
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. . . . .	122
4.17 Les séquences conçues par Proteus et les séquences naturelles représentées sous forme de logo pour les positions du cœur hydrophobe . . . . .	123
4.18 Les séquences conçues par Proteus et les séquences naturelles représentées sous forme de logo pour les positions du cœur hydrophobe . . . . .	124
4.19 Structure native Tiam1 avec les hydrophobes pour des $\delta$ 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. . . . .	126
4.20 Structure native Cask avec les hydrophobes pour des $\delta$ 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. . . . .	147
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. . . . .	147



# Liste des tables

2.1	Une partie des balises possibles dans le fichiers de configuration de Proteus	38
3.1	Selected MC and REMC protocols	59
3.2	Test proteins	61
3.3	Designed sequence quality measures	67
3.4	Tests with 10 designed positions	71
3.5	Tests with 20 and 30 designed positions	73
3.6	Designed and Pfam sequence entropies	74
3.7	Test proteins	77
3.8	Designed sequence quality measures	77
3.9	Tests with 10 designed positions	78
3.10	Tests with 20 and 30 designed positions	79
3.11	Designed and Pfam sequence entropies	80
4.1	Les groupes d'acides aminés utilisés pour l'optimisation des énergies de référence.	88
4.2	La sélection de domaines protéiques PDZ	92
4.3	E-value et pourcentage d'identité des alignements Blast native versus native pour nos séquences PDZ.	93
4.4	Sélection des homologues.	93
4.5	Similarité des séquences expérimentales homologues, pour les 8 protéines PDZ.	94
4.6	Les énergies de référence obtenues avec l'optimisation 6 protéines.	111
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.	112
4.8	Résultats Superfamily pour les séquences Proteus avec le modèle NEA.	113
4.9	Résultats Superfamily pour les séquences Rosetta	114

## ***Liste des tables***

---

4.10 Moyenne de l'exponentielle de l'entropie sur les ensembles des positions des protéines . . . . .	115
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. . . . .	119
4.12 Les énergies de référence obtenues avec l'optimisation sur 3 protéines. . . . .	120
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). . . . .	121
4.14 Pourcentage d'identité moyen à la séquence native . . . . .	121
15 Les tests avec cinq positions actives . . . . .	156
16 Les tests avec dix positions actives . . . . .	157
17 Les tests avec vingt positions actives . . . . .	158
18 Les tests avec trente positions actives . . . . .	159

# Abbreviations

<b>3D</b> tri-dimensionnel(le)	
<b>AMBER</b> Assited Model Building and Energy Refinement	
<b>BLOSUM</b> BLOcks of amino acid Substitution Matrix	
<b>CASA</b> Coulomb Accessible Surface Area	
<b>CHARMM</b> Chemistry at HARvard Macromolecular Mechanics	
<b>DEE</b> Dead-End Elimination	
<b>CPD</b> Computational Protein Design	
<b>GB</b> Born Généralisé	
<b>FDB</b> Fluctuating DIelectric Boundary	
<b>H</b> algorithme heuristique	
<b>MC</b> algorithme Monte-Carlo	
	<b>NEA</b> Naive Environnement Aproximation
	<b>PB</b> Poisson-Boltzmann
	<b>REMC</b> Replica Exchange Monte-Carlo
	<b>GMEC</b> "Global minimal energie cost"
	<b>Pfam</b> "Protein family databank"
	<b>backbone</b> (ou squelette) chaîne principale de la protéine
	$\frac{1}{2}RT$ avec R la constante des gaz parfait et T la température
	<b>DFBB</b> Depth-First Branch and Bound
	<b>EPT</b> Equivalence Preserving Transformation
	<b>MSD</b> Multistart Steepest Descent heuristic



# 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 de 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, on parle également du problème inverse du repliement. 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 prédition 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 résidues.
2. Un fonction d'énergie, qui permet d'évaluer la pertinence des conformations

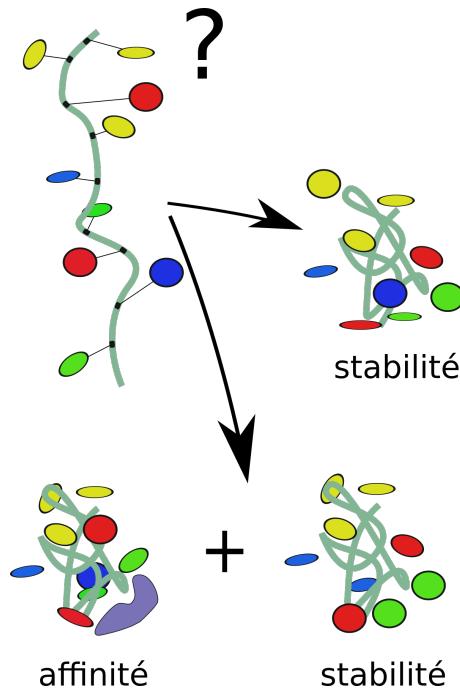


Figure 1.1 – Le CPD dans proteus

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'acides aminés, le nombre de polypeptide 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 positionnements dans l'espace 3D.

Ensuite, des variations sur ce principe ont été introduites, avec notamment l'introduction 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 à 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 établi que la chaîne latérale des résidus, sur un ensemble de protéines, adopte de façon préférentielle un petit ensemble de conformation (fig ??). Janin introduit alors le terme de « rotamère » pour désigner ces conformations. Il est alors possible de réduire l'espace continu des conformations des acides aminés à cet ensemble discret de rotamères. La plupart des méthodes de CPD utilisent cette discréttisation. Beaucoup de bibliothèques de rotamères ont été proposées dans la littérature scientifique. La plupart sont indépendantes du backbone. Mais il existe également des bibliothèques qui dépendent du squelette de la protéine voir [147,148]. Le nombre de structures de protéine utilisé est variable, La bibliothèque de Tufféry utilise 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 prédition 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 obtenu de nombreux succès , voir ???. Cependant, cette approximation peu avoir des conséquences 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  $\alpha$ ,  $\psi$  en introduisant des variations aléatoires 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 mouvement 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 prédition des mutants par rapport à des simllutations à squelette rigide.

### 1.1.2 l'état déplié

Lorsque la stabilité d'une protéine est évalué 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 hexaustive est difficile. Une approche simple consite 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 acide 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 modélisé 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  $E_{lié}$  l'énergie d'interactions des atomes éloignés d'au plus deux liaisons covalentes et  $E_{inbond}$  l'énergie des autres interactions. Détailons 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  $\theta_{ij}$  l'angle entre les liaisons i et j,  $\theta_{ij}^0$  l'angle optimal et  $k_{\theta,ij}$  la force du ressort.

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

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

où n est périodicité de la rotation,  $A_{i,n}$  est la constante de torsion,  $\Phi_i$  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  $a_2 - a_3$ .  $\Phi^0$  est la phase.

4. L'énergie de déformation des angles impropres  $E_{impr}$  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  $\Phi$  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 :

$$E_{\text{non liées}} = E_{\text{elec}} + E_{\text{vdw}} \quad (1.8)$$

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

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

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

2. L'énergie  $E_{\text{vdW}}$  pour l'énergie de Van der Waals, elle est due aux interactions électriques de faible intensité entre deux atomes, due au 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 :

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

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  $\epsilon$  qui varie en fonction de la distance interatomique. Il existe des fonctions d'énergie comportant des éléments relevant de la statique 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 type de solvant, ils ne seront pas abordé 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 molécule différentes.

Il est alors assez courant de considérer que l'énergie intramoléculaire n'est jamais modifiée, cela revient à considérer que les positions 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 qui 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,  $\epsilon_0$  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 initial 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'autre part, 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 coûteuse 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 coûteuses 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  $\Delta G_{solv}$ , elle est appelée l'énergie de solvatation.

On peut décrire  $\Delta G_{solv}$  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.  $\Delta G_{elec}$  l'effet électrostatique, qui correspond à la reorganisation des charges de la protéine dans le solvant
2.  $\Delta G_{VdW}$  est l'effet des forces de Van der Waals.
3.  $\Delta G_{cav}$  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 est 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  $\sigma_{t_i}$  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 enfui 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.

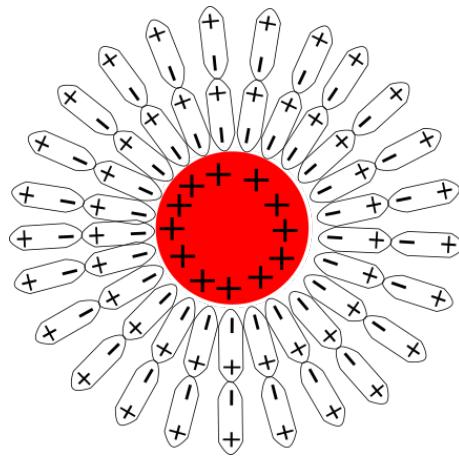


Figure 1.2 – organisation des dipôles des molécules d'eau autour d'un soluté sphérique chargé négativement à sa surface

### 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 intrasoluté, mais aussi les interactions électrostatiques entre groupes chargés de la protéine et solvant polarisé. Le continuum est munie d'une distribution de charge  $\rho_P$ , 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

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 Debye-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-Boltzman ; 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 proce 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 milieu de constante diélectrique  $\epsilon$  :

$$\Delta G_{elec} = -\frac{1}{8\pi} (1 - \frac{1}{\epsilon}) \sum_{i=1}^N q_i^2 \alpha_i \quad (1.20)$$

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

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

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  $\tau = \frac{1}{\epsilon_w - \epsilon_p}$ ,  $\epsilon_w$  étant la constante diélectrique du solvant,  $\epsilon_p$  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] :

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

Le calcul des rayons de Born  $\alpha_i$  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-calcul des énergies et le stockage de celles-ci, ce qui peut être problématique, par exemple dans les situations où le backbone est réfléchi, 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 constitue 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. Elles ont l'avantage d'être utilisables sur les espaces d'états non-discretisé, par exemple [154], les rotamères 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 besoins. 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 algorithmiques, il devient nécessaire d'imposer des contraintes sur la structure de la fonction d'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 parmi 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 :

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

avec  $N_i$  le nombre total de rotamère à la position  $i$ , et  $\beta = \frac{k_B T}{E}$ ,  $R$  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

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

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 energie propre et une énergie de paires.

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 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 utilisés alternativement, jusqu'à ce qu'il n'y ait plus de rotamère à éliminer. Il est alors possible d'utiliser une approche hexausicive 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 proposées : 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 rotamères.

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 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 possible 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 possible, 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

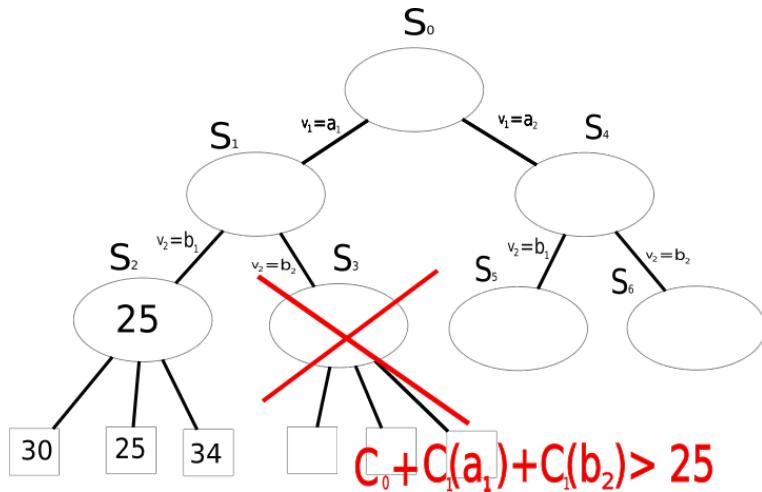


Figure 1.3 – Principe du l'algorithme du Depth-First Branch and Bound

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 coût de l'ensemble des couts unaires vers le coût 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 coût d'une fonction unaire vers les fonctions binaires impliquant la même valeur de variable transition \$CD\$. Une telle transformation permettant une augmentation du transfert total vers \$C\_0\$, CFN F.

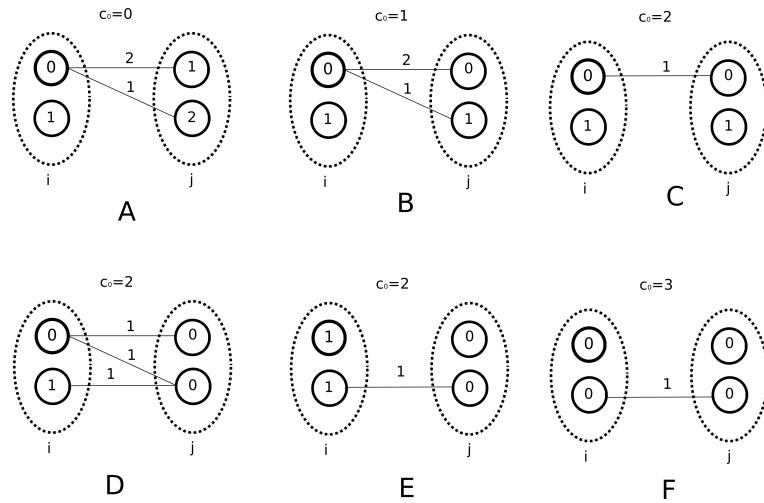


Figure 1.4 – Exemple de transformation EPT

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  $\pi$  : *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  $\pi$ .*

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 nécessite pas la connaissance de la constante de normalisation de  $\pi$ .*Cela constitue un attrait important parce qu'elles sont beaucoup moins coûteuses que les méthodes de type*

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  $x_{i-1}$ .*C'est typiquement*

On appelle processus stochastique, une suite de variable aléatoire  $X_{nn0}$ à valeurs dans  $E$ .*Une chaîne de Markov*

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

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

$$\mu_1 = \mu_0 \cdot P, \text{ avec } \cdot \text{ le produit matriciel. et}$$

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

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 :

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

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  $\pi$ .

Mais cela une 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.

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 :

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

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  $\pi^B$ , 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

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

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 sélection symétrique et ne violant pas 1.4.6 pour obtenir notre convergence.

Par la suite Hastings, généralise l'algorithme à une probabilité  $\text{sel}()$  non symétrique avec  $\text{acc}()$  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 1.43, 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 i de la trajectoire faire
3   choisir une proposition  $S'_i$  à partir d'une probabilité conditionnelle  $\text{sel}(.,S_i)$ ;
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). La méthode est également connue sous les noms « parallel tempering ». L'objectif est d'accélérer la convergence en permettant au processus stochastique de visiter plusieurs zones énergétiques simultanément. Ainsi, l'algorithme couple l'exploitation dans les basins de basses énergies , importante pour la recherche des solutions optimum avec, l'exploitation dans des zones de plus hautes énergies, ce qui facilite la sortie des minimum locaux.

On considère  $N$  simulations MC du système à  $N$  différentes températures. Ces répliques du système sont indépendantes les-unes des autres. On note  $T^m$  avec  $m = 1, \dots, N$  les différentes températures. Toutes ces températures sont différentes et il y a toujours exactement une replique à chaque température. Ainsi, nous nous plaçons dans un ensemble généralisé noté  $E^N$ , constitué des  $N$ -uplets  $(x^1, \dots, x^n)$  avec les  $x_i$  éléments de  $E$ , et  $1 < i < N$  indexant les répliques. On travaille alors avec une chaîne de Markov  $X_{t_1 < t}$ , avec  $X_t = (x_t^1, \dots, x_t^n)$  sur l'espace d'état  $E^N$ . On ajoute alors, un nouveau type de déplacement, celui qui consiste à intervertir au temps  $t$ , l'état  $x^i$  de la simulation à la température  $T^i$  avec l'état  $x^{i+1}$  à la température  $T^{i+1}$ . On a alors :

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

L'algorithme consiste alors à effectuer de façon itérative :

1. un ensemble de  $D$  déplacements, avec  $D$  une constante, de type MC
2. puis, une tentative de déplacement de type 1.48 est effectuée.

En effet, Pour que  $X$  respecte la balance détaillée il est nécessaire d'utiliser ce nouveau mouvement dans certains conditions. Il est donc associé à une probabilité d'acceptation  $acc_{swap}$  spécifique. Pour la déterminer nous reprenons la même démarche que pour le Monte-Carlo simple. Comme les repliques sont sans interactions, la distribution cible de la chaîne de Markov est égale aux produits des distributions cibles aux différentes températures :

$$\pi^B((x^1, \dots, x^n)) = \frac{1}{Z} \exp\left(-\sum_{i=1}^N E_{x^i}/kT^i\right) \quad (1.49)$$

En injectant cette expression de  $\pi^B$  et 1.48 dans ?? on a :

$$acc_{swap}(X_{t+1}|X_t) = \frac{\pi^B(X_t)}{\pi^B(X_{t+1})} = \frac{\exp\left(-\sum_{i=1}^N E_{x^i}/kT^i\right)}{\exp\left(-\sum_{i=1}^N E_{x^{i+1}}/kT^{i+1}\right)} = \exp(E_{x^j}/kT^j - E_{x^{j+1}}/kT^{j+1}) \quad (1.50)$$

$$\text{avec } \delta = (E_{x^j}/kT^j - E_{x^{j+1}}/kT^{j+1})$$

Ceci peut être satisfait par le critère de Metropolis :

si

$$\delta E > 0 \text{ alors } acc_{swap}(X_{t+1}|X_t) = \exp(-\delta) \quad \text{sinon } acc_{swap}(X_{t+1}|X_t) = 1 \quad (1.51)$$

On peut alors décrire l'algorithme comme en ??

```

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 ;

```

Il reste au simulateur à adapter le nombre de réplique, les températures utilisées et la fréquence des tentatives d'échange à sa problématique. La capacité de REMC à être exécuté sur des machines parallèles à entraîner sa grande popularité, en particulier dans le domaine de la modélisation moléculaire. Et son utilisation en physique, biologie, chimie, intelligence artificielle, etc augmente rapidement.

## 1.5 Les programmes CPD

## 1.6 Les succès

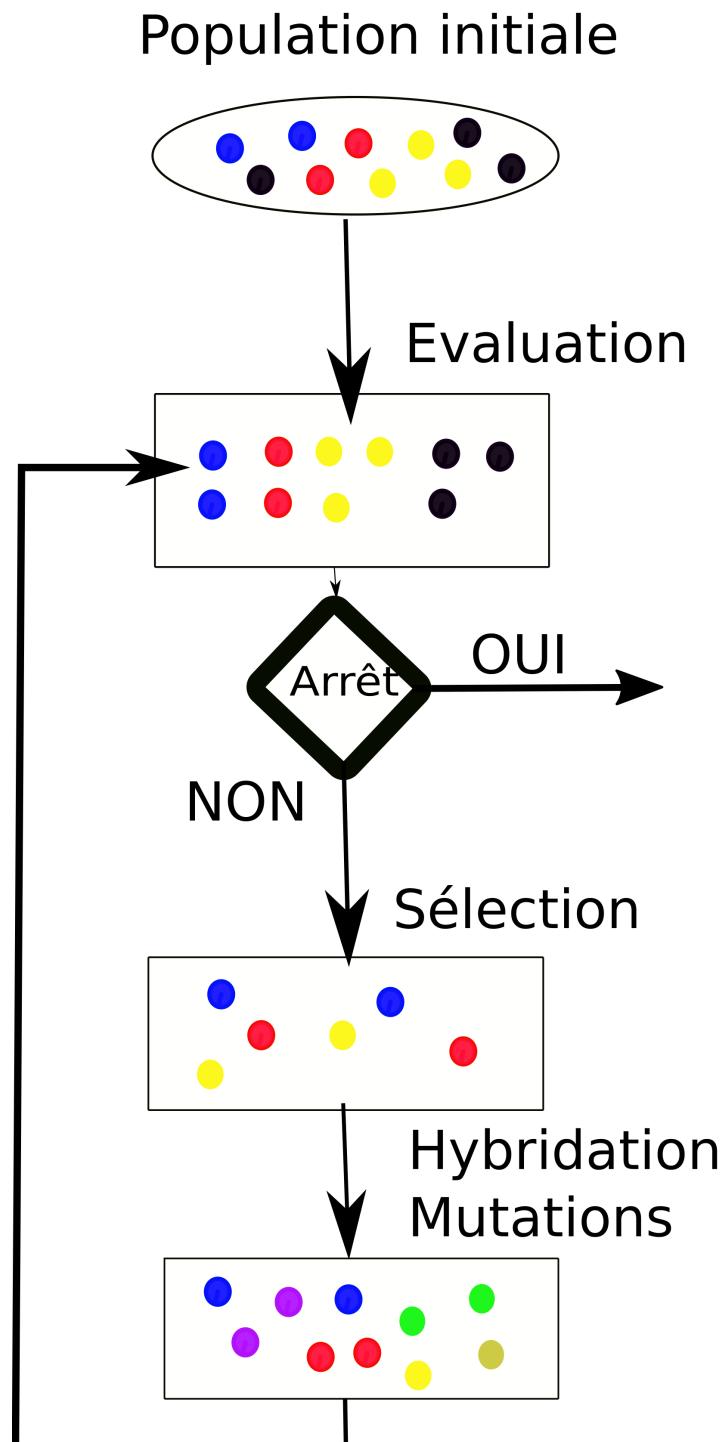


Figure 1.5 – L'algorithle génétique

## Chapitre 2

# Méthodes

### 2.1 Proteus

Proteus est constitué de deux parties :

1. un ensemble de scripts XPLOR, qui contrôle le calcul de la matrice d'énergie. (XPLOR est un programme de simulation moléculaire (??), disponible en téléchargement sur le site de l'université de Yale).
2. un programme écrit en C , que nous appelons « proteus » dans la suite de ce texte, qui explore l'espace de séquence/conformation

Le logiciel est flexible et largement configurable ; Il permet l'utilisation de plusieurs champ de force, de plusieurs modèle de solvant, et de plusieurs librairie de rotamer. La partie en C, proteus, propose plusieurs algorithme d'exploration comme l'heuristique, MC ou REMC .Le programme proteus permet de diviser le système en plusieurs groupes ou d'en dupliqué une partie , ceux-ci pouvant alors être combiné dans la fonction d'énergie afin de favoriser la stabilité de certains sous-ensembles du système ou certaines affinités. Plusieurs succès ont déjà été obtenu avec Proteus par exemple de redesign de la tyrosyl-tARN synthétase (??) ou sur la D-tyrosine ref, sur les calculs de  $pK_a$ ??.

Nous détaillons les points importants de Proteus.

Notre logiciel Proteus autorise le choix entre plusieurs type de fonction d'énergie.Dans le type le plus simple », est combinée l'approche de la mécanique moléculaire pour le traitement de la protéine et un modèle de solvant implicite de type CASA (voir ??). Deux types » sont également possible , dans lesquels le traitement du solvant est effectuée par un modèle Born Généralisé, avec soit une approximation NEA pour les rayons de solvatation, soit une approximation FDB. Comme il a été vu précédemment, avoir une fonction d'énergie décomposable par paire a plusieurs avantages, en particulier le calcul des énergies des paires de résidues avant l'étape d'exploration permet de réduire le calcul de l'énergie d'une séquence conformation pendant celle-ci à une somme d'énergies pré-

calculées. Cependant les termes GB et SA ne sont pas décomposable par paire. Il faut alors introduire de nouvelles approximation pour le permettre.Nous allons dans la suite comment ces problèmes sont résolues dans Proteus.

### **2.1.1 décomposition en paire du terme de surface**

Le terme surfacique présenté en ( ??) ce défini comme :

$$E_{solv}^= \sum_i \sigma_{t_i} A_i \quad (2.1)$$

avec  $A_i$  la surface accessible au solvant de l'atome  $i$ , et  $\sigma_{t_i}$  un coefficient d'hydrophobicité du type de l'atome  $i$ . Ce terme n'est pas décomposable par paire de résidu, parce que la surface enfouie d'une première chaîne latérale par une chdeuxième peut aussi être enfouie par troisième chaîne latérale. Pour rendre ce terme décomposable, nous utilisons la méthode de Street et al ( ??41]), dans laquelle la surface enfouie d'une chaîne est calculée à partir d'une somme sur les chaîne et les groupes du backbone voisins.Puis pour chaque groupe voisins, la zone de contact avec notre chaîne est calculée indépendamment des autres groupes. Ces zones sont sommées et un facteur de contraction est appliqué mimant l'élimination des zones comptée à plusieurs reprise.Des travaux précédents effectuée dans notre laboratoire ont montré qu'un facteur de 0,65 fonctionne bien (37, 40).

### **2.1.2 « Native Environnement Approximation » (NEA)**

Dans le terme GB de l'énergie de solvatation, le rayon de solvatation  $b_i$  approche la distance de l'atome  $i$  à la surface de la protéine, et donc c'est une fonction de la position relative à  $i$  de tous les atomes de la protéine.Et ce rayon n'est pas décomposable par paire.Pour qu'il ne devienne, Proteus utilise l'approximation « Native Environnement Approximation » ou NEA, dans laquelle le rayon de solvatation de chaque chaîne latérale est calculé avant l'étape d'exploration, en fixant tout le reste du système dans sa séquence native et sa conformation native[27,40].

### **2.1.3 « Fluctuating Dielectric Boundary » (FDB)**

Une nouvelle approximation du GB a récemment été introduite dans Proteus, toujours avec l'objectif, de transformer ce terme en en terme additif par paire.Elle exploite le fait que dans le GB, l'environnement dielectrique d'une paire de résidue est complètement caractérisé pour un petit ensemble de rayon de solvatation d'atome.Ces rayons sont eux-

mêmes sommes de paire sur les atomes de la protéines voir 14 16.La méthode s'appelle Fluctuating Dielectric Boundary » ou FDB et comporte les deux étapes suivantes :

La premiere consiste à définir un rayon de solvatation  $B_I$  pour chaque résidu  $I$  de la protéine.

On commence par définir une énergie propre à chaque paire de résidus I,J par la somme suivante :

$$E_{IJself} = \sum_{i \in I, j \in J} E_{ij} \quad (2.2)$$

puis l'énergie propre d'un résidu I :

$$E_{self} = \sum_J E_{ij} \quad (2.3)$$

Alors le rayon de solvatation moyen  $B_I$  est défini par :

$$E_I^{self} \stackrel{def}{=} \tau \sum_{i \in I} \frac{q_i^2}{2B_I}. \quad (2.4)$$

Nous avons, alors

$$\left( \sum_{i \in I} q_i^2 \right) \frac{1}{B_I} = \sum_{i \in I} \frac{q_i^2}{b_i}. \quad (2.5)$$

Donc,  $B_I$  est la moyenne harmonique des  $b_i$ , avec  $i \in I$ , pondéré par les charges au carré.

Il et alors possible de définir la contribution  $g_{IJ}$  de la paire de résidu I et J à l'énergie d'écrantage total  $\Delta$ , par :

$$g_{IJ} = \sum_{i \in I, j \in J} \tau q_i q_j \left( r_{ij}^2 + B_I B_J \exp[-r_{ij}^2/4B_I B_J] \right)^{-1/2} \quad (2.6)$$

Avec, la somme pour  $I = J$  excluant le cas  $i \neq j$ . On peut noter qu' aux distances fixes  $r_{ij}$ , et avec  $B = B_I B_J$ ,  $g_{IJ(B)}$  varie faiblement en fonction de B. Simonson et al (dans 54), propose alors d'approximer cette fonction par :

$$g_{IJ}(B) \approx c_1^{IJ} + c_2^{IJ} B + c_3^{IJ} B^2 + c_4^{IJ} B^{-1/2} + c_5^{IJ} B^{-3/2} \quad (2.7)$$

Les coefficients  $c_n^{IJ}$  peuvent alors être pré-calculé et stocké dans la matrice d'énergie, puisque les distances  $r_{ij}$  ne sont pas des variables pour un couple de chaîne latérale I,J

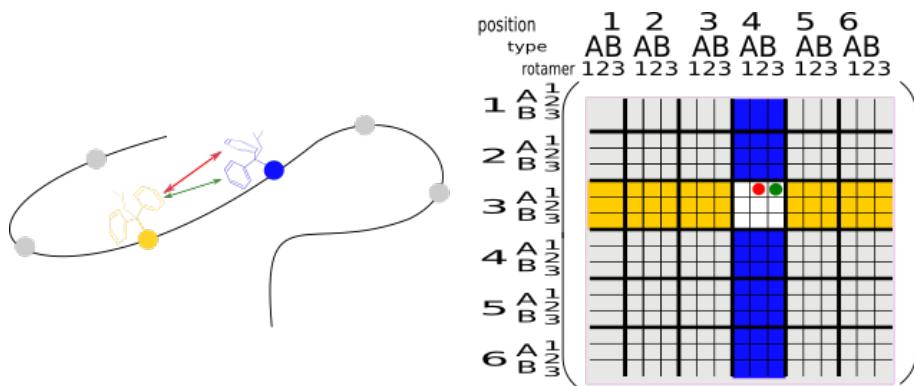


Figure 2.1 – **La matrice d'énergie** Sur cet exemple, un polypeptide de 6 résidues, chaque position posséde type d'acide aminés possibles et 3 rotamères possibles (2 pour le type A et 1 pour le type B). La matrice organise toutes les interactions de paires de chaîn latérales possibles. Les interactions impliquant le résidu numéro 2 sont dans la bande jaune de la matrice, les interactions impliquant le résidu numéro 3 sont dans la bande bleue. Les points rouge et vert correspondent aux interactions noté par les flèches rouge et verte à gauche (La matrice est symétrique , ici les interactions sont repräsentées qu'une seule fois).

donné. Comme les rayons de solvatation d'un résidu sont eux-mêmes des sommes sur les paires, le calcul de l'énergie GB à l'aide de 2.7 est maintenant décomposable par paire.

#### 2.1.4 La matrice d'énergie

Proteus utilise d'une part un backbone fixe, une espace discréte de rotamère et une fonction d'énergie décomposable par paires. Ces trois élément permettent de pré-calculer toutes les énergies interactions possibles. A cet ensemble d'interactions , il faut ajouter les interactions des résidues avec le backbone ceci pour chacun des rotamères possibles, pour constituer un ensemble complet de valeur énergétiques qui permet d'obtenir la valeur de la fonction d'énergie pour chaque sequence/conformation.Cet ensemble peut être organisée sans le format d'une matrice symétrique dans laquelle chaque couple de position dans la chaîne polypeptidique, apparait avec sa multiplicité de couples de rotamères possibles voir ??.

Le calcul de la matrice d'énergie, sont executer à l'aide du programme X-plor ; il se déroule de la façon suivante :

- Les atomes du squelette , c'est à dire les N,H,C<sub>α</sub>,C et O , sont fixés une fois pour toute. - Les atomes des chaînes latérales sont placés avec les angles dièdres issue de la bibliothèque de rotamères de Tuféry ( ??). - La fonction d'énergie qui utilise le champ de force Amber ?? de la forme :

$$E = E_{bonds} + E_{angles} + E_{dihedral} + E_{impr} + E_{vdw} + E_{Coul} + E_{solv} \quad (2.8)$$

avec  $E_{solv}$  pouvant être obtenu par le modèle CASA , le modèle GBNEASA ou GBFDBSA , correspondant respectivement à la formule ??,?? et ??, ceci selon la configuration de l'utilisateur.

### 2.1.5 l'état déplié

le  $\Delta\Delta G$  du chapitre précédent, s'applique entre l'état replié de la protéine et un état déplié. Nous avons donc besoin d'attribuer une énergie à l'état déplié.Cet état ne correspond pas à une structure précise pour une certaine séquence d'acide aminé. de même on peut considérer que pour un résidu donné la position les deux résidues immédiatement de doit pas impacter fortement l'énergie.Cela inspire une definition indépendante la structure telle que :

$$E_u(S) = \sum_i^N E(t_i) \quad (2.9)$$

avec  $E(t_i)$  une énergie du type d'acide aminé  $t_i$ .

Ces énergies est prise en entrée dans Proteus, et donc sa détermination ne fait pas encore partir des fonctionnalité intégré à notre logiciel.Il apparaît clairement dans la suite que la determination doit être fonction du système étudier. Nous détaillerons dans le chapitre PDZ, plusieurs méthodes dont de nouvelles et plusieurs exemple de determination qui seront exploitées.

### 2.1.6 déroulementement de la construction de la matrice

A partir un fichier PDB, une série de script XPLOR va préparer le système, ce système peut contenir un ligand. -1- L'utilisateur configure l'execution : Il determine :

1. un champ de force. Les champs AMBER ff99SB ou CHARMM toph19 sont tous les deux supportés.
2. un modèle de solvant parmi CA GBNEA GBFDB
3. un jeu de coefficients  $\sigma_i$  pour la pondération de la surface accessible au solvant dans le terme hydrophobe
4. l'ensemble des chaînes latérales autorisé à se déplacer

-2- Pour chaque résidu, un ensemble de conformation est défini à partir d'une librairie de rotamer. Cet ensemble est sauvegardé dans un fichier PDB par position.

-3- Les rayons de solvatation de Born sont calculé selon l'approximation GB demandé. Ces rayons sont sauvegardé dans un fichier dédié. -4- pour chaque rotamère de chaque

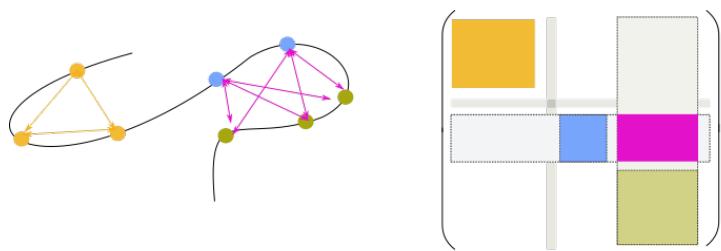


Figure 2.2 – Les principales structures « physiques » dans proteus

type, après une petite minimisation, est calculée ou lui seul peut se déplacer, l'énergie d'interaction avec le squelette. Elle est stocké dans un fichier. Ce sont les énergies de la diagonale de matrice. L'objectif de la minimisation est d'adapter le rotamère à son environnement natif. -5- Pour chaque paire de rotamère possible, comme pour la diagonale de ma matrice, une petite minimisation est effectuée avec uniquement la paire courante qui peut se déplacer. Puis les termes d'énergie d'interaction du couple sont calculés et enregistrés dans des fichiers.

Ensuite, les différents fichiers d'énergies seront lus par Proteus voir ?? pour construire la matrice.

### 2.1.7 les groupes dans proteus

### 2.1.8 modèle de données

### 2.1.9 Les entrées\_sorties

#### Le fichier de configuration

#### Les fichiers « backbone » et « pairwise »

#### Les fichiers de sortie

Type	nom	Description
methode d'exploration	Mode	determine le mode d'exécution, les valeurs possibles sont HEURISTIC, MONTECARLO, MEANFIELD et POSTPROCESS
nombre de pas	Trajectory_Length	la longueur de la trajectoire MC ou REMC
	Trajectory_Number	le nombre de trajectoire MC ou REMC
	Cycle_Number	le nombre de cycle en mode HEURISTIC
	Sequence_Pass_Number	le nombre maximum d'iteration sur la structure à chaque cycle(mode HEURISTIC)
fonction d'énergie	Optimization_Configuration	definition de la fonction d'énergie
	Group_definition	groupes d'energies et d'énergies d'interactions, ce sont les éléments de base de la fonction d'énergie
restrictions de l'espace de sequence/rotamer	Space_Constraints	restraint les états visités
paramètre du modèle	Temperature	attribue les températures aux marcheurs MC
Configuration	Random_Generator	Le générateur de nombre aléatoire de la « GNU Scientific Library »
	Rot_Proba	probabilité d'avoir un changement de rotamère à chaque pas
	Rot_Rot_Proba	probabilité d'avoir un double changement de rotamère à chaque pas
	Mut_Proba	...
	Mut_Mut_Proba	...
	Mut_Rot_Proba	... (ancienne version de Proteus)
	Position_Weights	probabilité de tirage de chaque position, lors de la première choix
Monte Carlo	Step_Definition_Proba	probabilité de changer un rotamère ou un type d'aa
	Neighbor_Threshold	à chaque pas les changement se font dans le même voisinage énergétiques. Cette balise définit la taille des voisinages.
Input/Output	Fasta_File	le nom du fichier produit par le mode POSTPROCESS
	Seq_Output_File	le nom du fichier de séquences produit par le mode HEURISTIC ou MONTECARLO
	Energy_Output_File	le nom du fichier d'énergie produit par le mode HEURISTIC ou MONTECARLO

Table 2.1 – Une partie des balises possibles dans le fichiers de configuration de Proteus

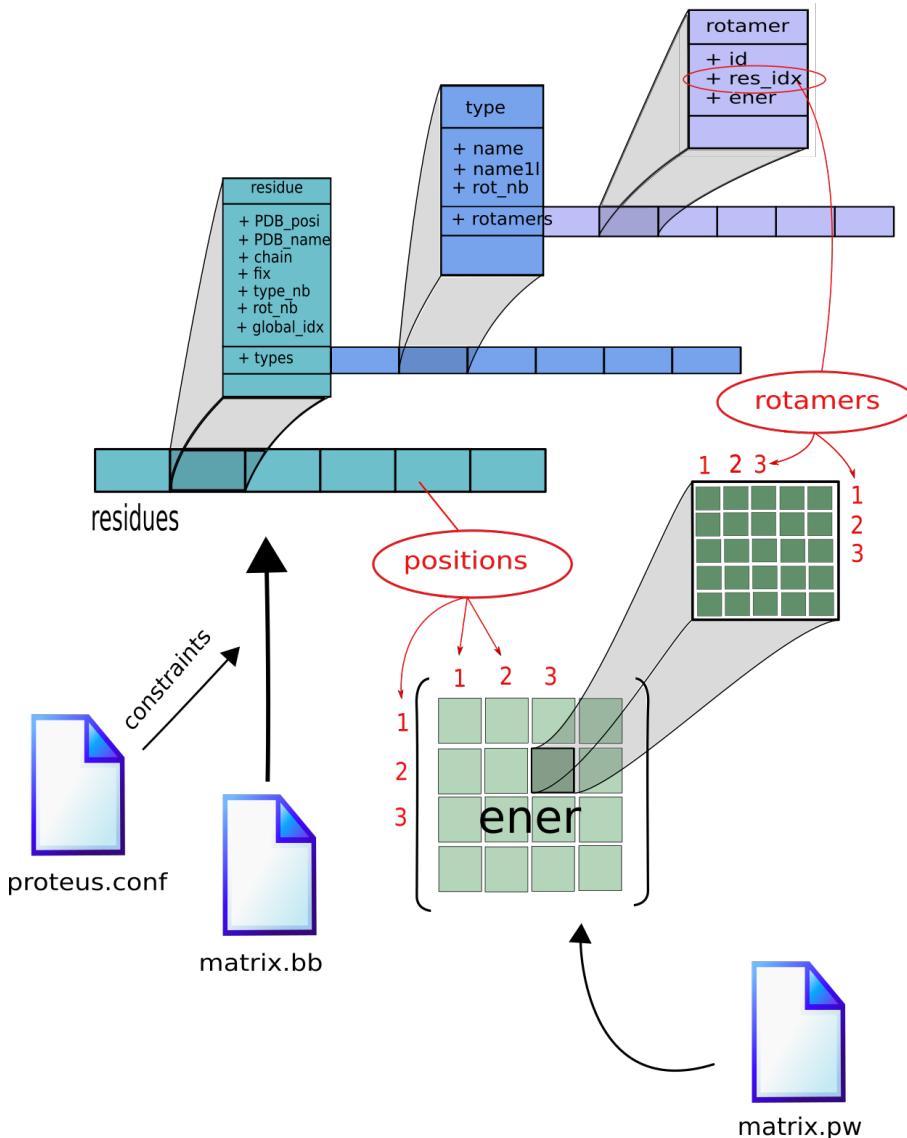


Figure 2.3 – Les principales structures « physiques » dans proteus

## 2.2 Description des tests de comparaisons d'algorithme

Nous cherchons maintenant à déterminer les performances et les qualités des différents algorithmes de proteus. Pour évaluer les différents algorithmes de proteus, comme pour leur établir un paramétrage, nous effectuons des séries de tests. Grâce à l'algorithme de type toulbar2 il est possible d'obtenir la séquence/conformation qui possède la plus haute énergie de dépliement. Cela constitue une information importante qui va nous servir d'élément de comparaison. Le facteur temps est également un élément déterminant. Il est dans certain cas limitant, nous ne savons pas à l'avance quand toulbar2 termine. Et il apparaît d'emblée illusoire d'espérer voir ce programme converger dans toutes les

## 2.2. Description des tests de comparaisons d’algorithme

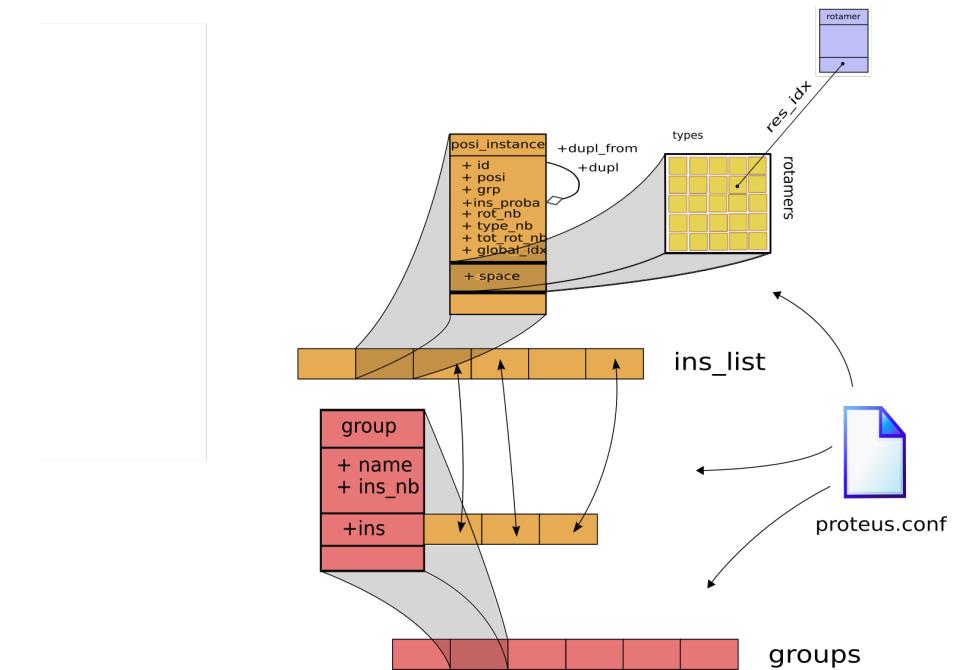


Figure 2.4 – Les principales structures « logiques » dans proteus

situations intéressantes dans un temps raisonnable. D’autres métriques qui caractérisent les séquences d’acides aminés de meilleures énergies obtenues seront également utilisées pour les évaluations et pour les paramétrages.

Dans la suite, on appelle «position active», une position pour laquelle, tous les types d’acides et tous les rotamères de chaque type d’acide aminé sont autorisés, au court de la recherche de proteus. On désigne «séquence/conformation» une séquence d’acides aminés munie à chaque position d’un rotamère (le backbone étant de toute façon fixé). Tandis que le terme simple «séquence» sans plus de précision désigne une séquence d’acides aminés.

### 2.2.1 Sélection des positions

Nous cherchons à déterminer les performances et les qualités des différents algorithmes d’exploration de proteus. Pour évaluer les différents algorithmes, comme pour leur établir un paramétrage, nous effectuons des séries de tests. Grâce à l’algorithme de type toulbar2 il est possible d’obtenir la séquence/conformation qui possède la plus haute énergie de déploiement. Cela constitue une information importante qui va nous servir d’élément de comparaison. Le facteur temps est également un élément déterminant. Il est dans certain cas limitant, nous ne savons pas à l’avance quand toulbar2 termine. Et il apparaît d’emblée illusoire d’espérer voir ce programme converger dans toutes les situations intéressantes

## Chapitre 2. Méthodes

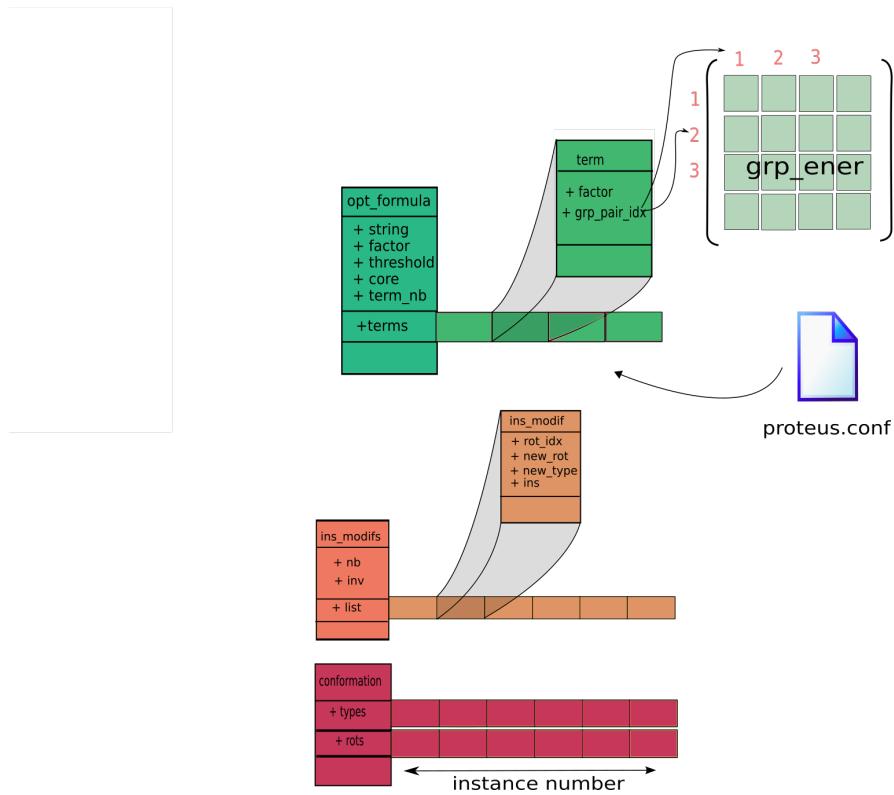


Figure 2.5 – Les principales structures « dynamiques » dans proteus

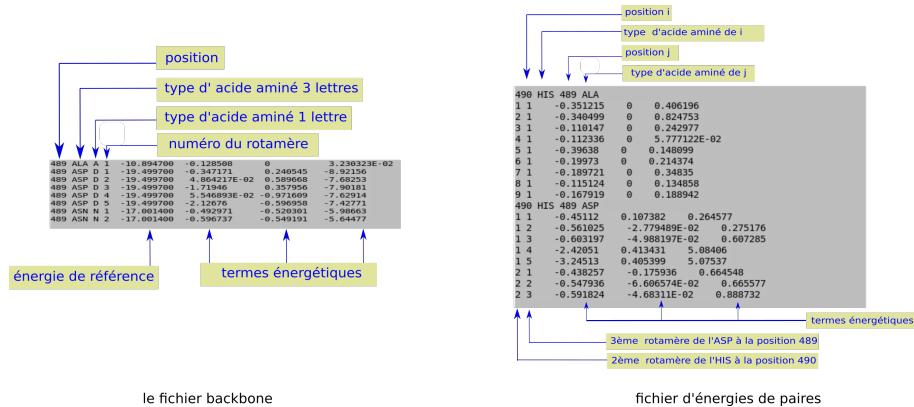


Figure 2.6 – Les fichiers en entrée

dans un temps raisonnable. D'autres métriques qui caractérisent les séquences d'acides aminés de meilleurs énergies obtenues seront également utilisées pour les évaluations et pour les paramétrages.

Dans la suite, on appelle «position active», une position pour laquelle, tous les types d'acides et tous les rotamères de chaque type d'acide aminé sont autorisés, au court de la recherche de proteus. On désigne «séquence/conformation» une séquence d'acides aminés

## 2.2. Description des tests de comparaisons d'algorithme

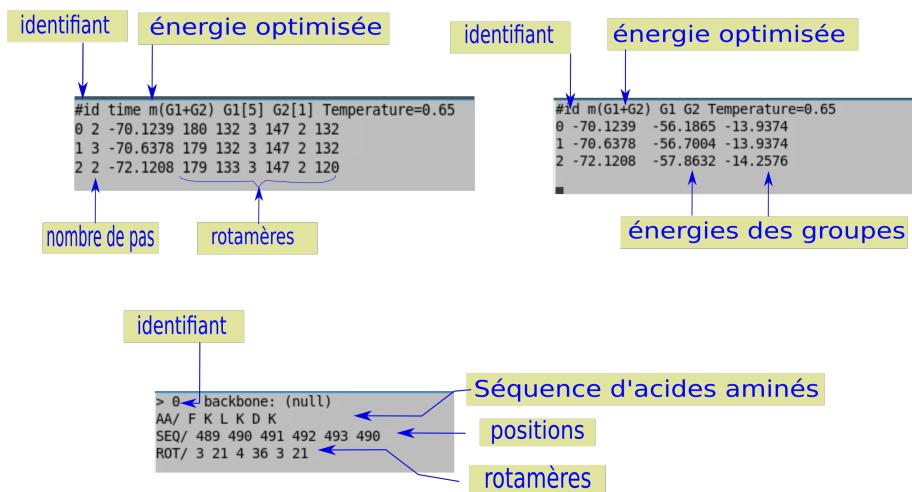


Figure 2.7 – Les fichiers en sortie de proteus

munie à chaque position d'un rotamère (le backbone étant de toute façon fixé). Tandis ce que le terme simple «séquence» sans plus de précision désigne une séquence d'acides aminés.

Les tests sont répartis en deux ensembles :

1. un ensemble de tests où toutes les positions de la séquence sont actives (cela correspond aux situations de design complet de protéines)
2. un ensemble de tests où le nombre de positions actives est gardé sous contrôle de façon à maîtriser la taille de l'espace d'exploration

**Ensemble «Tout actif»** Pour le premier ensemble de tests, la totalité de la matrice d'énergie est exploitée et pour chaque position l'espace d'exploration correspond à l'espace d'état déclaré dans le fichier ".bb". C'est-à-dire que tous les types de résidu et tous les rotamères sont possibles à chaque position. Comme l'espace des séquences/conformations à explorer est gigantesque, nous ne faisons pas de tentatives de recherche du GMEC par méthode exacte.

Nous effectuons des recherches avec les algorithmes suivants :

- heuristique, noté H par la suite ;
- Monte-Carlo, noté MC ;
- «Replica Exchange», noté RE) ;

**L'ensemble «nombre d'actifs limité»** L'ensemble «Nombre d'actifs limité» est composé de six groupes de tests avec un nombre de positions actives fixe défini de la façon suivante :

## **Chapitre 2. Méthodes**

---

1. aucune position active
2. une position active
3. cinq positions
4. dix positions
5. vingt positions
6. trente positions

Lorsqu'une position n'est pas active, l'acide aminé de la position est fixé en utilisant l'acide aminé de la séquence native. La chaîne latérale est, elle, laissée libre. Il n'y a donc jamais dans nos tests de position où l'état est complètement fixé.

Le groupe «aucune position active» n'est constitué que d'un test par algorithme pour chaque protéine. Il y a donc neuf tests par algorithme. Ce sont les tests pendant lesquels la séquence d'acides aminés est fixe et correspond à la séquence native de la protéine.

Pour les tests avec une seule position active, comme des temps de calcul le permettent, nous décidons d'être exhaustifs : Toutes les positions sont testées, il y a alors huit cent quatre tests par algorithme. Pour tous les autres groupes de tests (cinq,dix,vingt et trente positions actives), cinq tests sont effectués par protéine, c'est-à-dire quarante-cinq tests par algorithme.

**le choix des positions actives** Pour définir complètement les tests, il reste maintenant à décrire le choix des positions actives pour les groupes de numéro trois jusqu'à six. Il y a peu d'intérêt à tester des situations avec des positions actives sans interaction entre-elles. En effet, s'il existe une position active P dont chaque résidu est sans interaction avec tous les résidus possibles des autres positions actives, déterminer le meilleur état pour P est proche du test du groupe 2 avec P comme position active. Toutefois, cela n'est pas exactement la même question, parce que les positions actives différentes de P peuvent influencer la position de la chaîne latérale de positions inactives qui à leur tour peuvent influencer l'état de P. Ainsi, le choix des positions actives ne se fait non pas par tirage aléatoire, car le risque d'obtenir des positions avec peu d'interactions est trop grand. Il se fait sous contrainte d'interaction.

**positions en interactions** Pour cela, nous utilisons la notion de voisinage de proteus. Elle se définit de la façon suivante : Deux positions P et Q sont en interactions s'il existe un rotamère  $r_P$  de P et un rotamère  $r_Q$  de Q tels que :

$$|E(r_P, r_Q)| > S_{Vois}$$

## **2.2. Description des tests de comparaisons d'algorithme**

---

avec  $S_{Vois}$  un seuil donné par l'utilisateur à la configuration de proteus (voir chap. ?? pour les détails).

Alors on appelle «n-uplet en interaction» la donnée de n positions avec  $n \in \{5,10,20,30\}$  et d'un seuil  $S_{Vois}$  tels que pour toute paire de positions (P,Q) du n-uplet, P et Q sont en interactions.

**choix des positions actives** Pour définir les positions actives,nous exécutons proteus en mode verbeux, sans effectuer d'optimisation.Pour cela, il existe plusieurs façons de procéder, ici nous utilisons le mode Monte-Carlo avec une trajectoire de zéro pas. Ces exécutions produisent en sortie standard la liste des voisins pour chaque position au seuil donnée en paramètre. Pour chacune des neuf protéines, nous exécutons proteus avec  $S_{Vois}$  égal à dix, cinq et un à tour de rôle ; trois listes de voisins sont obtenues. Ensuite,un script dédié recherche dans ces listes, les n-uplets en interaction, en partant de la liste de voisins au sens le plus fort, c'est-à-dire dix, vers celle au sens le plus faible (0.1).La recherche s'arrête lorsque cinq n-uplets au moins sont trouvés.

Nous obtenons quarante-cinq n-uplets pour le groupe à cinq (respectivement dix, vingt et trente ) positions actives pour un seuil  $S_{Vois}$  égal à dix (respectivement dix, un et un). Les positions actives de tous les tests sont en annexe ??).Pour chaque n-uplet, un fichier de configuration de proteus est créé dans lequel la balise <Space\_Constraints> fixe les positions inactives en utilisant le type d'acide aminé présent dans la séquence native.

### **2.2.2 Définition de protocole comparable**

Nous voulons comparer les algorithmes très différents. Un algorithme peut garantir l'obtention du minimum global en énergie (GMEC) si l'exécution se termine, mais ne garantit pas qu'elle se termine. Un autre permet un contrôle très fin du temps d'exécution sans garantie du GMEC, et d'autres enfin ont des objectifs plus large que la seule obtention du GMEC. Mais le GMEC reste le meilleur point de commun. Nous allons donc y concentrer une part importante des comparaisons.

Nous devons noter également que l'obtention du GMEC est théorique, en pratique nous n'avons pas de preuve que le code de l'algorithme exact que nous utilisons n'a pas de bogue. Cependant,nous mettons de côté cette éventualité et dans toute la suite GMEC désigne aussi bien le minimum global en énergie que le résultat de toulbar2 lorsqu'il se termine. Le Monte-Carlo et le «Replica exchange» possèdent de nombreux paramètres de configuration, ce qui rend l'ensemble des protocoles possibles très grand. Se pose alors la question de l'optimisation du protocole. L'objectif fixé ici,n'est pas la recherche d'un protocole optimal pour chacun des tests, mais d'évaluer, avec les tests, un protocole optimisé par algorithme.

## **Chapitre 2. Méthodes**

---

Nous allons alors dans un premier temps, recherche les meilleurs paramétrages pour le Monte-Carlo et le «Replica Exchange» sur l'ensemble de tests «tout actif». Puis, sur la base des résultats obtenus, les protocoles seront fixés pour effectuer les comparaisons sur l'ensemble «tout actif» et celui à «nombre d'actifs limité». Le programme toulbar2 possède aussi de nombreuses options. Deux paramétrages différents seront utilisés.

Pour rendre les protocoles comparables, le temps d'exécution maximum est fixé à vingt-quatre heures pour tous les exécutions. Toulbar2 donne sa meilleure séquence/conformation en dernier, il n'y a donc pas post-traitement nécessaire. C'est également le cas pour le Monte-Carlo à condition de configurer l'impression de la trajectoire avec la balise *Print\_Threshold = 0.* dans le fichier de configuration. Pour le "Replica Exchange" et l'heuristique, un tri des séquences selon l'énergie est nécessaire. Mais il n'y a pas beaucoup de séquences :

1. L' Heuristique fournit une séquence/conformation à chaque cycle.
2. Le "Replica Exchange avec *Print\_Threshold = 0* produit autant de fichiers de séquences/rotamères que de marcheurs. Chacun ne contenant pas plus de quelques dizaines de séquences/rotamères.

Nous pouvons donc négliger la durée du tri dans le temps total d'exécution.

**Protocole heuristique** Pour l'algorithme heuristique, il n'y a dans notre situation qu'un seul paramètre à renseigner : le nombre de cycles à effectuer. Quelques essais préliminaires sur la plus grosse protéine (Table ??) avec toute les positions actives, montre que la version utilisée de proteus peut effectuer jusqu'à environ 110000 cycles sur nos machines de calculs en l'espace de vingt-quatre heures. Ainsi, le protocole H est défini comme le protocole qui utilise le mode heuristique de proteus et qui effectue cent dix mille cycles. Sont également définis les variantes H-, H+ et H++ comme des protocoles plus courts ou plus longs à facteur entier près (Table ??). Par ailleurs, certaines comparaisons de l'heuristique avec le Monte-Carlo ont été faites avec une version précédente du programme proteus. Ce protocole sera noté h. Il diffère aussi de H par le fait que l'option d'optimisation du compilateur Intel utilisé est -O2 contre -O3 pour H.

**Protocoles Monte-Carlo** On distingue deux ensembles de protocoles Monte-Carlo. Dans le premier, les noms sont de la forme "mc\*". Il rassemble les protocoles utilisés pour le paramétrage du Monte-Carlo. Le second est constitué des protocoles utilisés lors des comparaisons.

Les éléments à paramétrier pour l'algorithme Monte-Carlo sont les suivants :

1. la température

## **2.2. Description des tests de comparaisons d'algorithme**

---

2. le nombre de pas (avec le nombre de trajectoires et la longueur de trajectoire )
3. Le seuil de voisinage
4. Les probabilités de changements de la séquence/conformation

Ce qui représente un ensemble de protocoles trop grand pour une approche exhaustive. Pour l'essentiel, nous allons faire varier les paramètres un par un, en prenant comme point de départ un protocole qui rend le comportement de marcheur Monte-Carlo «proche» de l'heuristique.

La température est le paramètre principal du Monte-Carlo, c'est elle qui contrôle le taux d'acceptation du critère de Metropolis. Alors, la première étape de cette optimisation va consister à faire varier la température , entre 0.001 et 0.5 , en conservant les autres paramètres fixés (protocoles de mc0 à mc5). Le nombre de pas total effectué est le produit de deux paramètres, le nombre de trajectoires et la longueur de trajectoire. Les protocoles mc1b et mc2b testent l'effet d'une augmentation du nombre de pas. Tandis que mc2c et mc2d testent l'effet de la variation du nombre de trajectoires par rapport à la longueur. Le protocole mc2e s'intéresse aux probabilités de changement de la trajectoire. Il y a cinq balises dans proteus qui contrôle ces changements :

- <**Prot**> donne la probabilité de modifications de rotamère à une position.
- <**Prot\_Prot**> donne la probabilité de modifications de rotamère à deux positions.
- <**Mut**> donne la probabilité de modifications de type de résidu à une position.
- <**Mut\_Prot**> donne la probabilité de modifications de rotamères à deux positions.
- <**Mut\_Mut**> donne la probabilité de modifications de type de résidu à deux positions.

La table ?? donne les probabilités utilisées par ces cinq paramètres dans l'ordre de la liste précédente.

Enfin, mc4b se distingue des autres par un seuil de voisinage plus grand ((Table ??)).

**Protocoles "Replica Exchange"** L'algorithme «Replica Exchange» (RE) est une extension du Monte-Carlo. Les paramètres d'un protocole RE sont ceux d'un protocole Monte-Carlo plus trois autres :

- le nombre de marcheurs
- la température pour chaque marcheur
- la période de «swap», c'est-à-dire la période (en nombre de pas) à laquelle le test de Hasting sur l'échange de température est effectué.

Pour avoir des exécutions en parallèle avec au plus un marcheur par cœur du processeur, nous limiter nos tests à quatre ou huit marcheurs. La distribution des températures est

## **Chapitre 2. Méthodes**

---

un élément déterminant dans le comportement des marcheurs, car c'est elle qui pilote en grande partie le taux d'acceptation des échanges de températures. Nous suivons l'idée proposée par Kofke de lui faire suivre une progression géométrique ( $\frac{T_i}{T_{i+1}} = C$ , avec C une constante) [??]. Ceci garantie alors que le taux d'acceptation d'échange entre  $T_i$  et  $T_{i+1}$  soit égale pour tout nos i. De plus, nous souhaitons centrer approximativement, nos distributions sur la température ambiante (environ 0.6 kcal/mol). Dans toute la suite, les températures et les énergies sont exprimées en kcal/mol.

Voici les températures pour le RE quatre marcheurs :

- 10, 1, 0.1 et 0.01
- 2, 1, 0.5 et 0.25
- 1, 0.5, 0.25 et 0.125

,et celles pour le RE huit marcheurs :

- 3 , 2 , 1.333 , 0.888 , 0.592 , 0.395 , 0.263 et 0.175
- 10 , 3.16 , 1 , 0.316 , 0.1 , 0.0316 , 0.01 et 0.00316

Ici les protocoles ne se font qu'avec une seule trajectoire par marcheur. Et la contrainte du temps de calcul se comprend comme vingt-quatre heures de calculs cumulées sur tous les marcheurs. Ainsi les longueurs de trajectoire sont définies pour le RE à quatre marcheurs comme le quart d'une trajectoire MC, pour le RE à huit marcheurs comme le huitième.

**Distribution des énergies en fonction des températures** Regardons plus en détail le comportement du programme pour l'algorithme "Replica Exchange". Notre implémentation de cet algorithme consiste à modifier la température de certains marcheurs à certains moments voir ?? . Pour examiner la trajectoire à une température donnée t ( c'est à dire le graphe dans l'espace les séquences/conformations selon les pas du marcheur lorsqu'ils sont à la température t), nous découpons les trajectoires des marcheurs selon la température. Puis nous collons les segments de trajectoires de même température en respectant l'ordre des pas. Comme les changements de températures se font de façon synchrone , c'est-à-dire un échange entre deux marcheurs, nous obtenons autant de trajectoires selon la température qu'il y a de marcheurs et ces nouvelles trajectoires sont de mêmes longueurs que les trajectoires obtenues par proteus. La figure 2.8 représente la distribution en énergie des trajectoires selon la température sur le test utilisant le protocole RE8b1 sur la protéine 1A81. Nous observons une courbe en cloche pour chaque température, retrouvant ainsi l'allure d'une distribution de Boltzman. L'espace de recouvrement entre deux courbes consécutives, représente la partie de la distribution dans laquelle les échanges de température vont être concentrés, parce que cette zone recouvre les situations où la température peut augmenter sans dégradation de l'énergie. En effet, le critère de Metropolis-Hastings peut

## 2.2. Description des tests de comparaisons d'algorithme

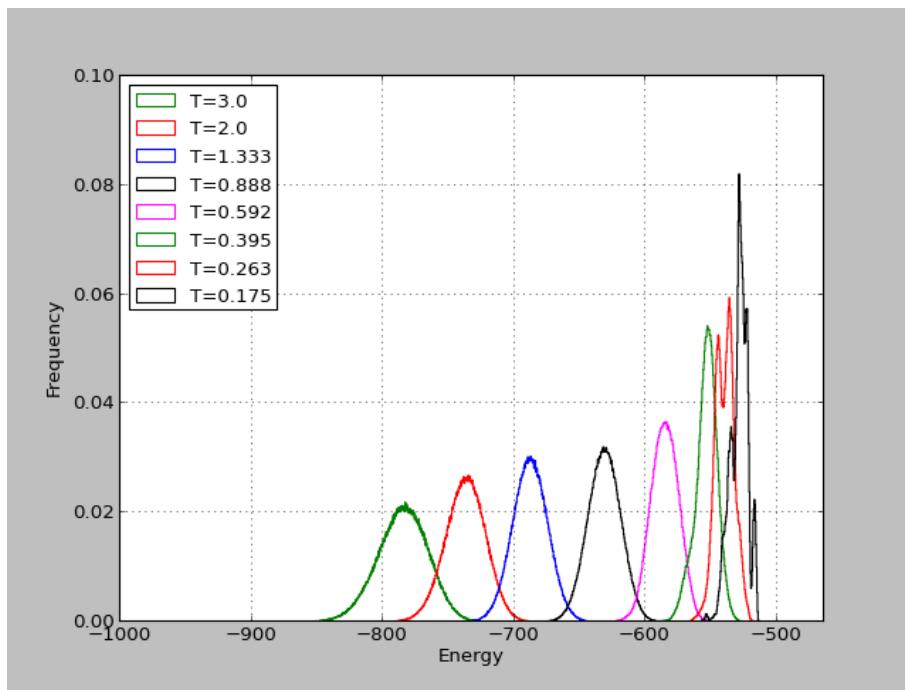


Figure 2.8 – Distribution des énergies selon la température (1A81 protocole RE8b1).

accepter un réchauffement de marcheur (respectivement un refroidissement) si son énergie diminue (respectivement, augmente), mais uniquement avec une probabilité qui décroît exponentiellement en fonction de la différence d'énergie. À l'inverse si les recouvrements des distributions consécutives sont trop importants le marcheur peut difficilement s'éloigner de la plage d'énergie d'une température. Le graphique nous confirme que le choix d'une plage de température entre 3.0 et 0.175 pour les protocoles huit marcheurs rend possibles les échanges de températures et permet également l'exploration d'une vaste zone d'énergie.

**Trajectoires dans l'espace des températures** Un critère de performance souvent utilisé est la proportion de changements de températures effectivement acceptés. Pour examiner ces changements, nous représentons la trajectoire dans l'espace des températures pour quatre tests sur la protéine 1A81. Il s'agit des tests effectués avec deux protocoles à quatre marcheurs, le RE4a et le RE4b et deux protocoles à huit marcheurs, le RE8a2 et le RE8b1, voir figure 2.9. Les deux graphiques quatre marcheurs représentent seulement la moitié des trajectoires de 1,5 million de pas, tandis que les graphiques huit marcheurs représentent la totalité de trajectoires de 750 millions de pas. Comme prévu, le protocole RE4a, qui possède des températures plus écartées, brasse assez mal les températures. En particulier, le marcheur w4 ne quitte presque pas la température la plus froide, et ce malgré une période de swap assez faible et un nombre d'échanges effectifs assez important.

## Chapitre 2. Méthodes

---

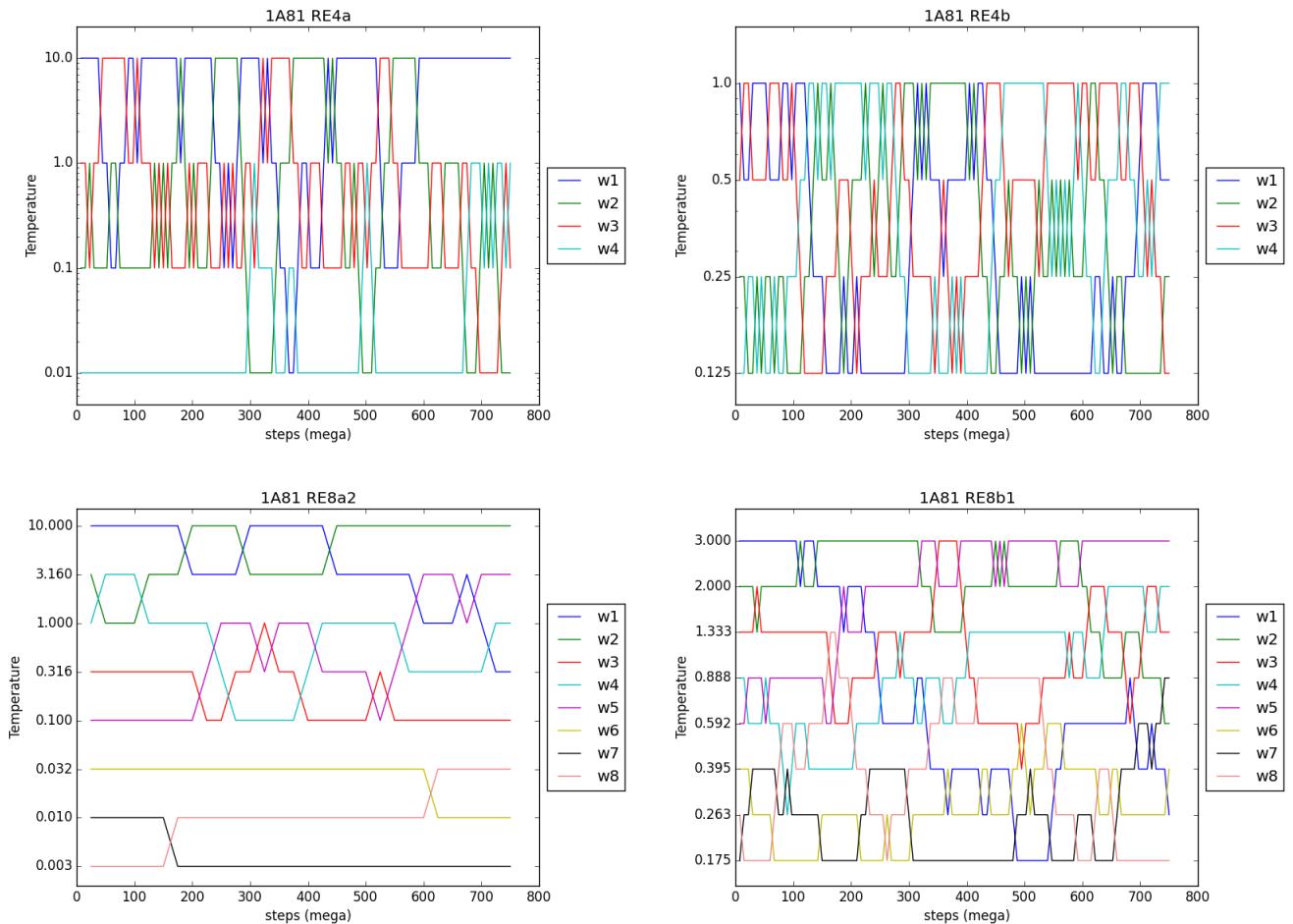


Figure 2.9 – Variation de la température pendant la trajectoire de chaque marcheur (cas exemple de protocoles).

On voit que pour RE4b, protocole avec les températures plus proches , la répartition des échanges est nettement plus uniforme. Le fait de double le nombre de températures pour le même intervalle que RE4a ne suffit pas à résoudre le problème et les trois marcheurs les plus froids en début de trajectoire pour le RE8a2 restent froids. Les bons résultats obtenus pour RE4b sont confirmés pour RE8b1 qui utilise également des températures proches.

**Protocoles Toulbar2** Après avoir converti nos matrices au format «wcsp» grâce à un script dédié,nous pouvons utiliser toulbar2. Le protocole de recherche du GMEC est le suivant : L'exécutable toulbar2 de version 0.9.7.0 est lancé avec les options « -l=3 -m -d : -s», ce qui correspond au paramétrage conseillé dans la documentation CDP [??]. Si l'exécution se termine en moins de vingt-quatre heures, le protocole est achevé. Sinon le programme est arrêté et une seconde version (la 0.9.6.0) est lancée avec les options «-l=1

## **2.2. Description des tests de comparaisons d’algorithme**

---

-dee=1 -m -d : -s». Au bout de vingt-quatre heures si le programme n'est pas terminé, il est arrêté. La dernière séquence/conformation imprimée en sortie est collectée. Le choix de la seconde version et du paramétrage fait suite à une discussion avec monsieur Seydou Traoré.

Toulbar2 offre également la possibilité de fournir la liste des séquences/conformations dont l'énergie est comprise entre celle qui correspond au GMEC,  $E_{GMEC}$  et une autre  $E_{upper\_bound}$  donnée en paramètre. Pour utiliser cette fonctionnalité nous utilisons le paramétrage : «-d : -a -s -ub= $E_{upper\_bound}$  ». Cependant, il s'avère que cette utilisation peut utiliser une quantité de mémoire vive importante. Alors, pour éviter tout plantage de nos machines, la mémoire que toulbar2 peut allouer est limité à 30 Go.

### **2.2.3 Outils d'analyse des données**

**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 plausible.

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.

**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

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

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.

**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 :

$$Id(A_S, i) = \frac{\sum_{1 < j < m} \mathbb{1}(s_i^j, n_i)}{m} \times 100, \text{ avec } m \text{ le nombre de séquences de } A_S.$$

**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

## **Chapitre 2. Méthodes**

---

séquence native. Mais cela n'est pas notre seule 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 n'utiliserons l'alignement dit « seed » qui se base sur un petit ensemble de membres représentatifs de la famille et l'alignement « full », plus large, qui est généré par modèle de Markov caché à partir de l'alignement «seed». Les alignements correspondent pour nous aux familles PF00017 (domaine SH2), PF00018 (domaine SH3) et PF00595 (domaine PDZ).

**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 la matrice BLOSUM62 (BLOcks SUbstitution Matrix) [?] qui est construite à partir de blocs d'alignement très conservés (ici plus de 62% d'identités). 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.

**similarité d'un ensemble à une famille 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 blastpgp est utilisée avec comme database (paramètre -d ) l'alignement Pfam et comme séquence en entrée ( paramètre -i ) 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.

## Chapitre 3

# Comparing stochastic search algorithms for computational protein design

Ce chapitre reprend l'article publié dans « Journal of Chemical Theory and Computation » ( ??).

## 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 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  $\sigma_E$ , we should enumerate all the possible sequences/structures within one or two  $\sigma_E$  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  $\sigma_E$  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 ??.

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  $\delta E$  (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

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

$\Delta E_M$  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.

### 3.2. Methods

---

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 ??? :

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

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

### Chapitre 3. Comparing stochastic search algorithms for computational protein design

---

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 :

$$\begin{aligned} acc(t \rightarrow t') &= e^{-\beta(\Delta E_f - \Delta E_u)} \frac{n_f(t)}{n_f(t')} = e^{-\beta(\Delta E_f - \Delta e_u)} \frac{n_f(t)n_u(t')}{n_u(t)n_f(t')} \quad \text{if } \Delta E_M > 0 \quad (3.8) \\ &= 1 \quad \text{otherwise} \end{aligned} \quad (3.9)$$

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,  $\Delta E_M$ .

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  $\beta_i$ ,  $\beta_j$  are the inverse temperatures of the two walkers and  $\Delta E_i$ ,  $\Delta E_j$  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.

Table 3.1 – Selected MC and REMC protocols

name	walker temperature(s) $kT$ (kcal/mol)	trajectory length (steps)	move probabilities <sup>a</sup> rot ; mut ; mut+rot ; mut+mut ;	walker swap periodicity <sup>b</sup>
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	$10^6$

<sup>a</sup>Probabilities 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. <sup>b</sup>The interval between attempts to exchange states between two walkers (using a Metropolis test).

### 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  $\sim 100.000$  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,  $\epsilon$  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 ;  $\sigma_i$  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  $\sigma_i$  values (cal/mol/Å<sup>2</sup>) : 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 3.7. 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).

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				

### 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 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 ?.

### **Chapitre 3. Comparing stochastic search algorithms for computational protein design**

---

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}, an {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

### 3.3. Results

is typically similar between methods. Table 3.8 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.

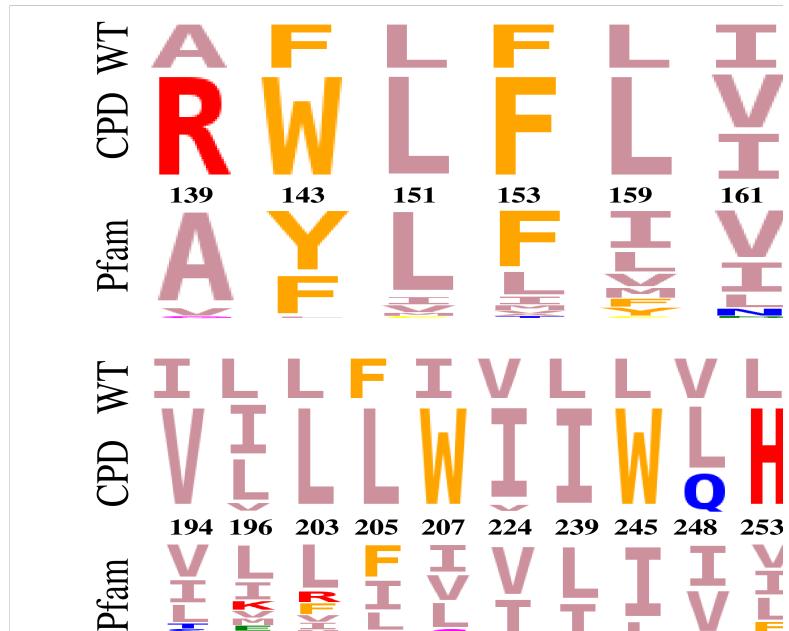


Figure 3.1 – Sequence logos for the core region of two designed proteins : 1CKA (SH3) and 2BYG (PDZ). The low energy CPD sequences are compared to the sequences of the full Pfam collection of experimental sequences. Positions shown correspond to the hydrophobic core of each protein ; residue numbers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

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

### **Chapitre 3. Comparing stochastic search algorithms for computational protein design**

---

members of the correct superfamily *and* family, with match lengths and E-values given in Table 3.8. Thus, our designed sequences are largely similar to experimental ones. Sequence identities to wildtype (including 1A81) are 31% on average (Table 3.8), similar to earlier studies with the same energy function ?? .The good agreement with experiment is also illustrated by computing similarity scores between the designed sequences and sequences from the Pfam database. For the protein core region, the similarity is similar to that between experimental sequences, as shown in Fig.1. Notice that we use here a simple CASA solvent model, since our focus is not the quality of the designed sequences, but the performance of our search algorithms. With a more sophisticated Generalized Born solvent model and a more highly optimized unfolded state model, sequence quality would be even better (manuscript in preparation). Representative sequence logos for the protein core are shown in Fig. ??, illustrating the agreement between designed and experimental sequences. For the SH3 protein 1CKA, the design mostly recovers the native sidechain type, or a homologous type that is common in other natural sequences from Pfam. Exceptions include position 170, where we obtain I or V, whereas the native type is W ; however, L and V are also occasionally found in Pfam. At position 139, we sample R instead of the native A or V, found in Pfam ; however, the hydrophobic part of the designed Arg sidechain is homologous to A and V, while its ionized tip actually sits outside the hydrophobic core, at the protein surface. Finally, at position 143, we sample W, which is homologous to Y and F found in Pfam. For the PDZ domain 2BYG, agreement with the native and Pfam sequences is similar, with a few departures : at positions 207 and 245, we sample W instead of the natural types I, L, V, P ; at position 253, we sample H, compared to mostly I, L, V, and sometimes F in Pfam ; at position 265,we always sample Y, whereas the Pfam types are diverse (and mostly hydrophobic).

For the same PDZ domain, the lowest energy sequence/structure combination is shown in Fig. ??, superimposed on the Xray structure ; only the hydrophobic core sidechains are shown, for clarity. Most of the native/designed sidechains overlap very well, although the double mutation (F205L, L245W) leads to some local repacking. For the 1A81 SH3 domain, the lowest energy sequences are not recognized by Super-family, but if we assay sequences that are 812 kcal/mol above the GMEC, about 10% are correctly recognized by Superfamily as SH3 sequences. Similarly, if the top 10,000 sequences produced by the heuristic algorithm are tested, 57% of them are recognized by Superfamily as SH3 family members. The heuristic sequences are also 812 kcal/mol above the GMEC (see below). This protein illustrates the imperfect parameterization of our energy function, with the consequence that higher energy sequences can actually be more realistic, in some cases,

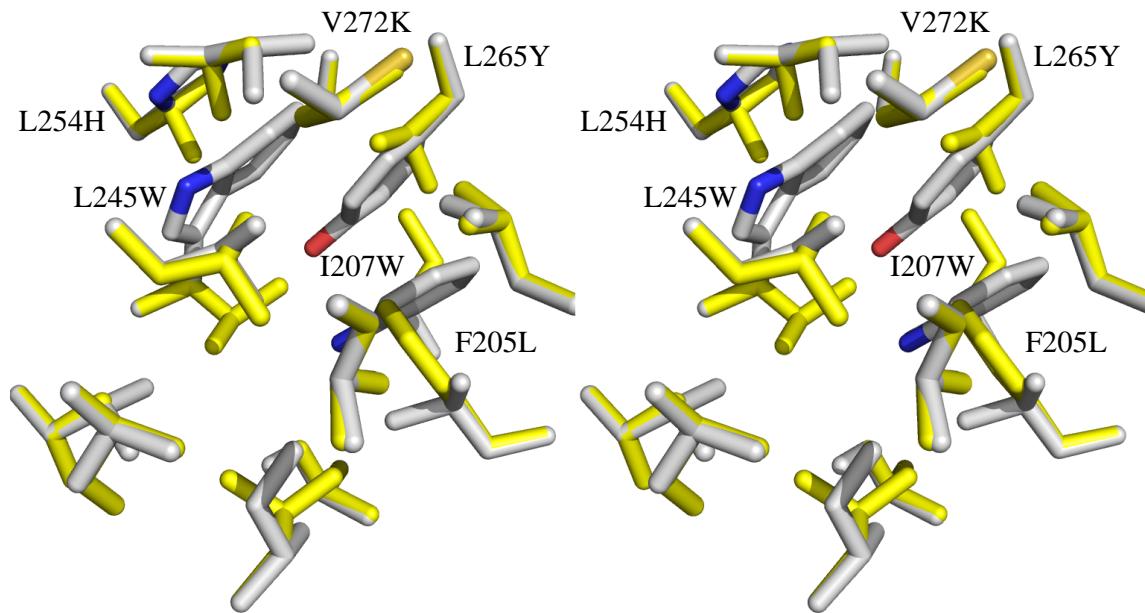


Figure 3.2 – The lowest energy sequence/structure combination is shown for the 2BYG PDZ domain (stereo view ; selected sidechains only ; white sticks), superimposed on the Xray structure (yellow sticks) ; the sidechains shown are the ones in the sequence logo, Fig.2. The main mutated positions are labelled ; other positions are either not mutated or mutated within the ILV group. Figure produced with pymol [72]

than lower energy ones. Thus, the signiance of the GMEC is only as good as the energy function.

### 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 ressources 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

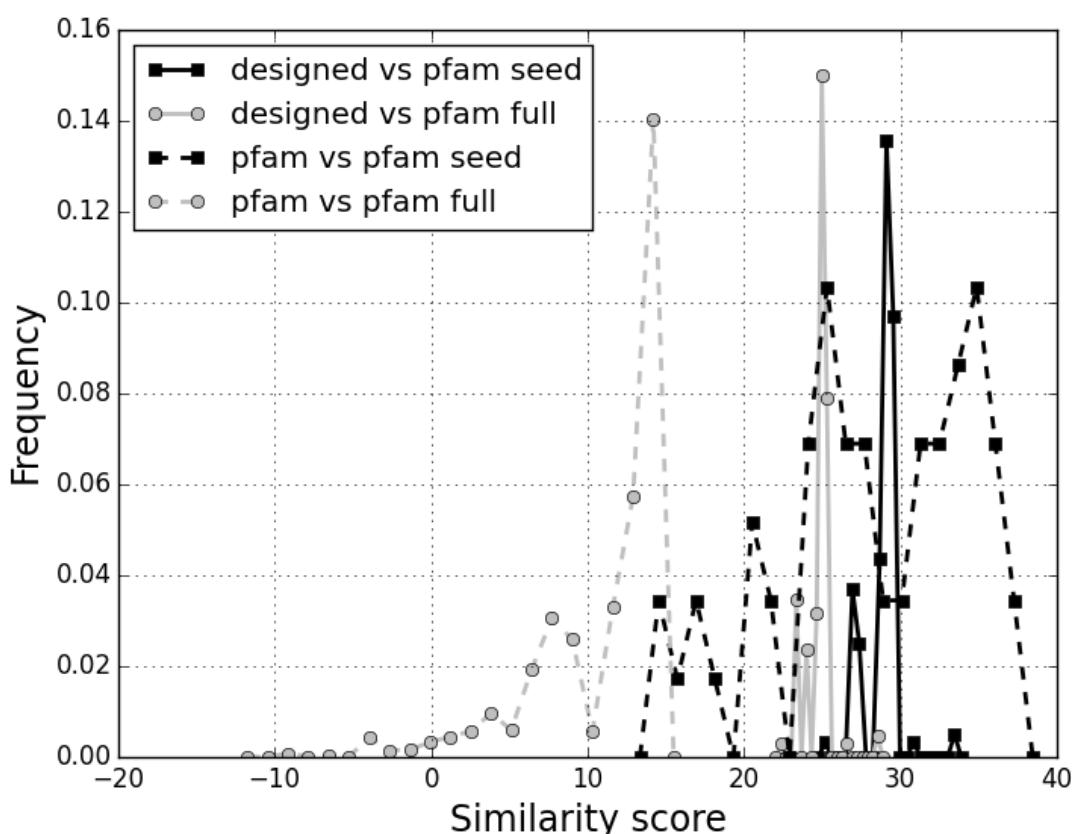
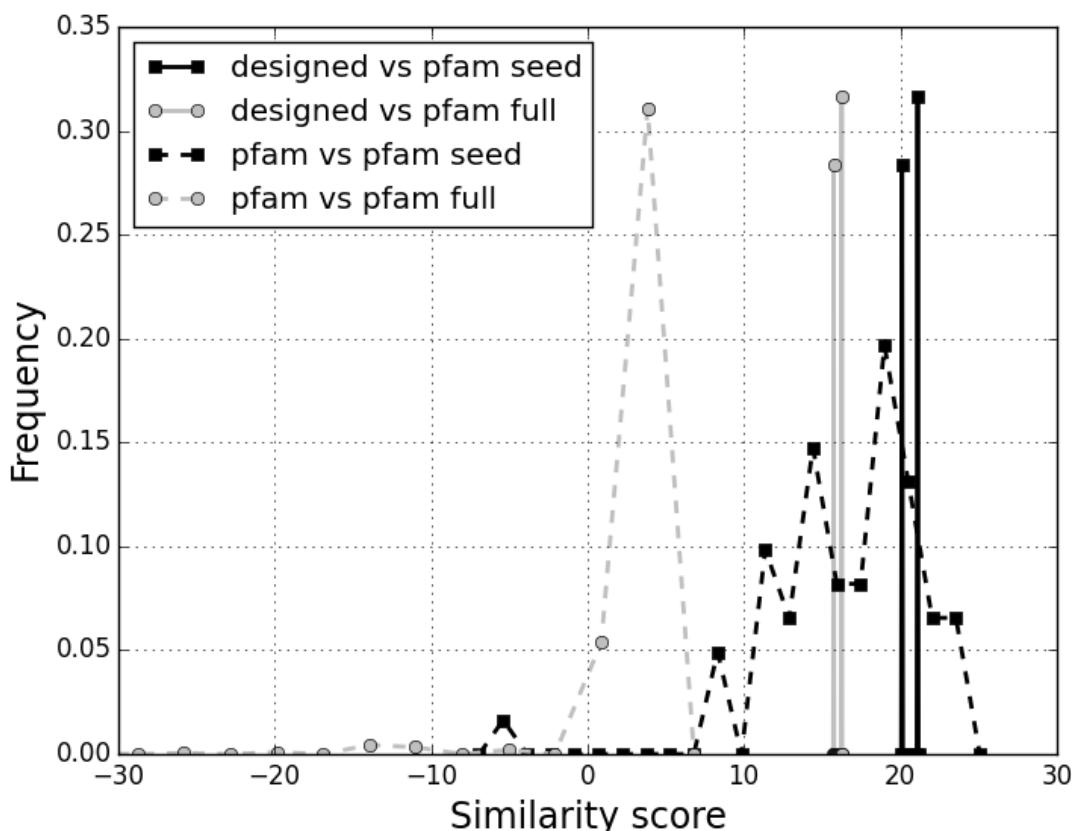


Figure 3.3 – Histogram of Blosum40 similarity scores to Pfam sequences for the core region of two designed proteins : 1ABO (SH3) and 1BM2 (SH2). The similarity between Pfam sequences is also shown, considering either the Pfam seed alignment or the much larger full alignment.

### 3.3. Results

---

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%

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. 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. 3.4 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. 3.5 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. 3.5. The lowest mean error

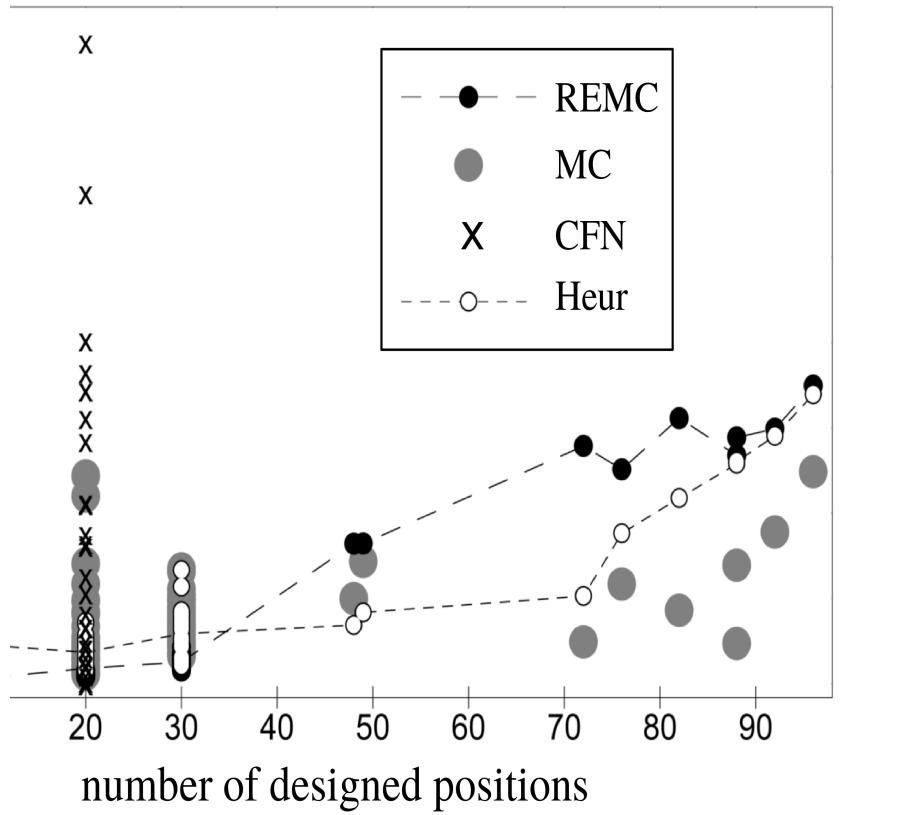


Figure 3.4 – Run times for different test calculations and search methods. CPU times per core are shown; only the REMC calculations use multiple cores (OpenMP parallelization). CFN results are only given for 20 designed positions or fewer. REMC was only done for the larger calculations. Results are shown 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.

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. ???. With zero positions, only

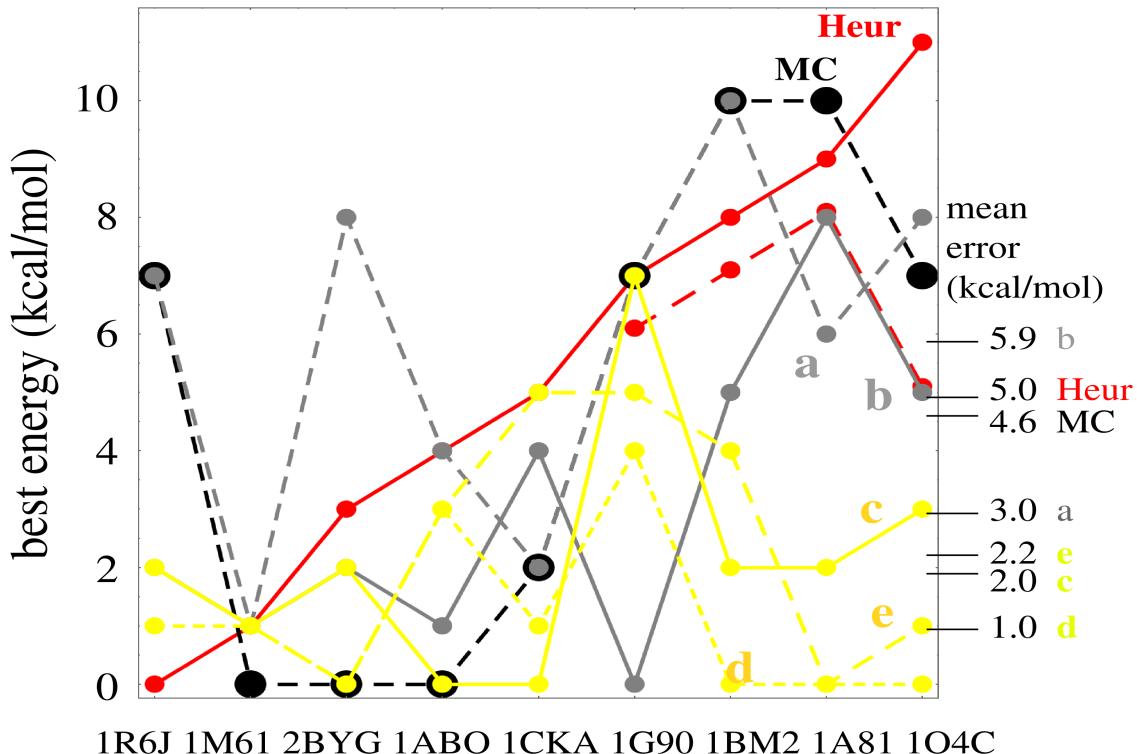


Figure 3.5 – Comparison between selected heuristic, MC, and REMC protocols for whole protein design. The best energy obtained with each protocol is shown for the nine test proteins. Zero represents the overall best energy for each protein. The mean “error” is indicated on the right for each protocol (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.

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. ??), with run times of a few minutes.

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

### ***Chapitre 3. Comparing stochastic search algorithms for computational protein design***

---

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. ?? and Table 3.9. 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 speed, for 10-position design, all the methods were comparable (a few hours per run on average).

### 3.3. Results

Table 3.4 – Tests with 10 designed positions

rotamers <sup>a</sup>	length <sup>b</sup>	Protein	CFN <sup>c</sup>	Heur. <sup>d</sup>	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	

<sup>a</sup>Total number of rotamers available to the system. Each designed position can explore 206 rotamers; the others explore about 10 rotamers each. <sup>b</sup>Total protein length (number of Gly+Pro in parentheses). <sup>c</sup>gmec indicates the GMEC was successfully identified. <sup>d</sup>For 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.

#### 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. ?? and Table 3.10. 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,

### **Chapitre 3. Comparing stochastic search algorithms for computational protein design**

---

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. ?? and Table 3.10. 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

### 3.3. Results

Table 3.5 – Tests with 20 and 30 designed positions

Protein	20 positions					30 positions			
	CFN	Heur.	MC	REMC	mutations <sup>a</sup>	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 3.9. gmec\* indicates the more aggressive protocol. <sup>a</sup>Between CFN/Heur.

### **Chapitre 3. Comparing stochastic search algorithms for computational protein design**

---

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 3.11 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.

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).

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. 3.6 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).

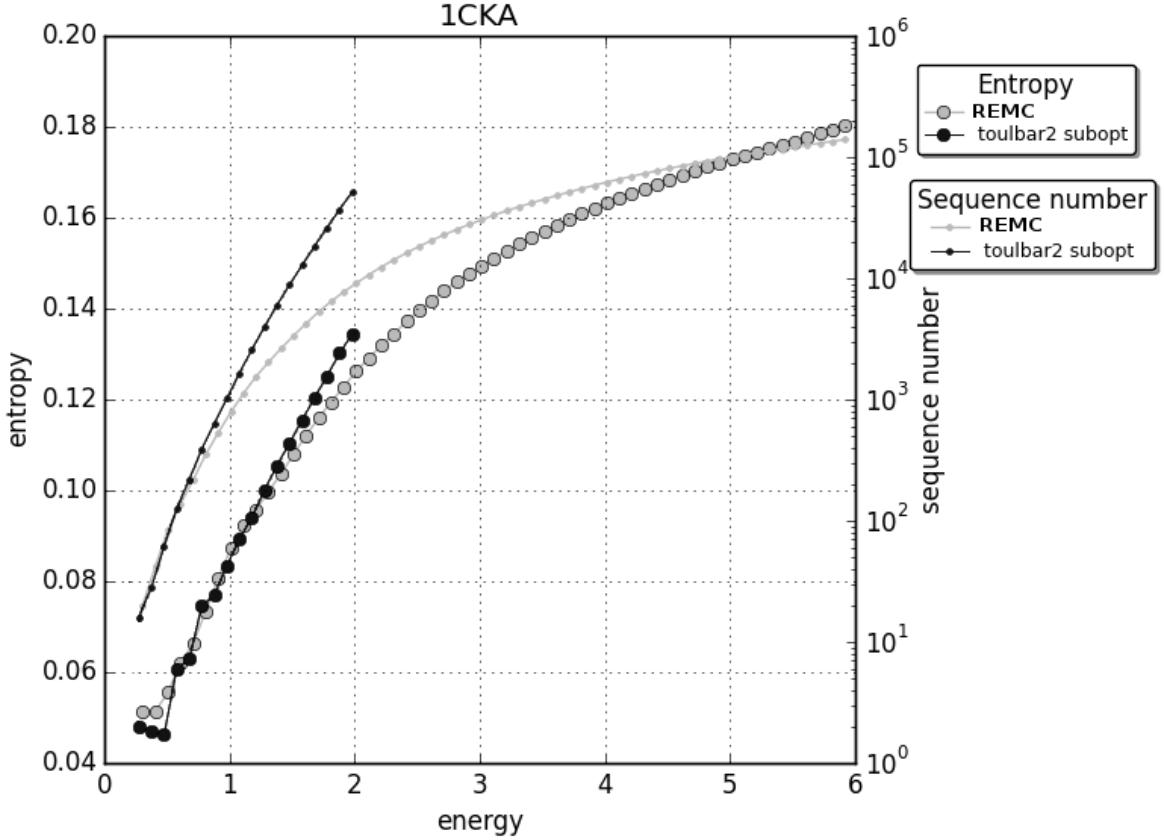


Figure 3.6 – Sequence entropy  $S(E)$  and number of states  $N(E)$  within a given energy range  $E$  above the GMEC for the 1CKA SH3 domain. All positions were allowed to vary except Gly/Pro. The entropy (large dots) is a single position sequence entropy, Eq. (4.4.7), averaged over all the variable 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.

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

### 3.4. Conclusions

---

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 3.7 – 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 3.8 – 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%

Table 3.9 – Tests with 10 designed positions

rotamers <sup>a</sup>	length <sup>b</sup>	Protein	CFN <sup>c</sup>	Heur. <sup>d</sup>	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	
		1CKA 5	gmec	0.2859	3.2525	0.
2508	57(8)	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	
		1O4C 1	gmec	0.	0.1121	
		1O4C 2	gmec	0.	0.1046	
3037	104(8)	1O4C 3	gmec	0.	0.1519	
		1O4C 4	gmec	0.	0.1545	
		1O4C 5	gmec	0.	0.1753	
		1R6J 1	gmec	0.	2.4022	0.3986
		1R6J 2	gmec	0.	1.0398	0.3049
2773	82(10)	1R6J 3	gmec	0.	0.0106	
		1R6J 5	gmec	0.	0.0162	
		2BYG 1	5.7485	0.	0.0337	
		2BYG 3	gmec	0.	0.0833	
2888	97(15)	2BYG 4	gmec	0.	0.2149	

<sup>a</sup>Total number of rotamers available to the system. Each designed position can explore 206 rotamers ; the others explore about 10 rotamers each. <sup>b</sup>Total protein length (number of Gly+Pro in parentheses). <sup>c</sup>gmec indicates the GMEC was successfully identified. <sup>d</sup>For 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 3.10 – Tests with 20 and 30 designed positions

Protein	20 positions					30 positions			
	CFN	Heur.	MC	REMC	mutations <sup>a</sup>	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 3.9. gmec\* indicates the more aggressive protocol. <sup>a</sup>Between CFN/Heur.

Table 3.11 – 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).

## 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  $\Delta E^f$  et  $\Delta E^u$  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  $\alpha$  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  $\delta E$  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  $\delta E$  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  $\delta E$  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  $\delta E$  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,  $\epsilon_w$  et  $\epsilon_p$  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  $i$ .  $\sigma_i$  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  $\sigma_i$  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  $\delta E_t^r$  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  $\delta E_t$  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.

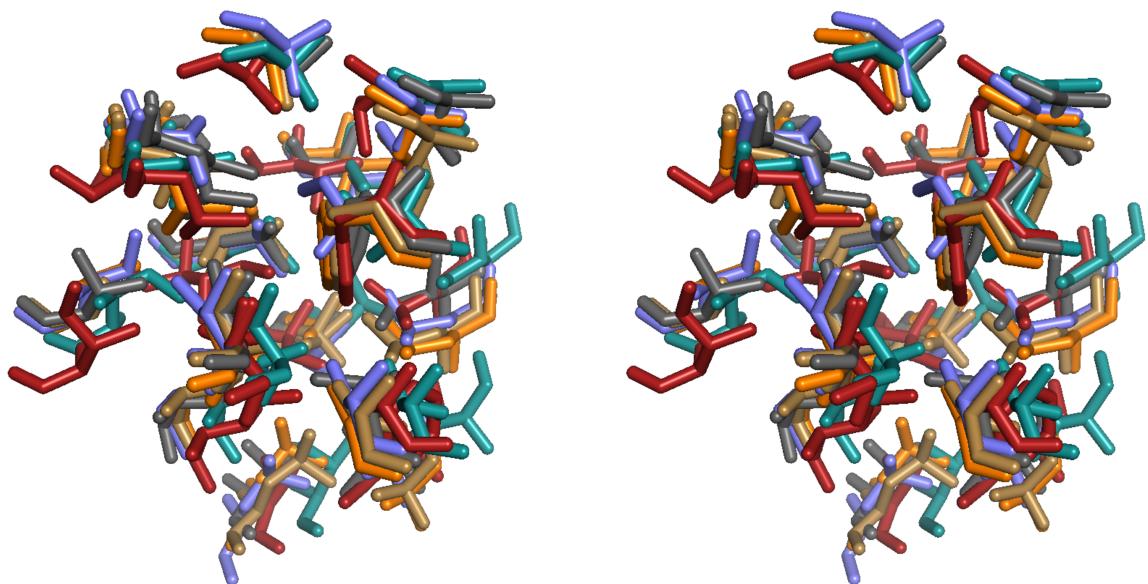
Protein	NHREF	INAD	GRIP	Syntenin	DLG2	PSD95	Cask	Tiam1
NHREF	326	64	15	15	59	112	49	1
INAD	64	221	56	-9	88	107	25	9
GRIP	15	56	378	24	65	87	90	39
Syntenin	15	-10	24	311	-26	22	42	-18
DLG2	59	88	65	-26	325	110	24	22
PSD95	112	107	87	22	110	325	66	21
Cask	49	25	90	42	23	66	308	37
Tiam1	1	10	39	-18	22	21	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.LLAGDRIVEVNGENVEKETHQQVVSRI  
 1IHJ GELIHMVLDKTGKSFVICIVRGEVKDSPNTKTTGIFIKGIVPDSPAHLCGRLKVGDRLSLNGKDVRNSTEQAVID  
 1N7E GAIYYTVELKR.YGGPLGITISGTEEP.....FDPIISSLTKGGLAERTGAIHGDRILAINSSSLKGKPLSEAI  
 1R6J GAMDPRTITMHKDSTGHVGFIK.....GKITSIVKDSSAARNG.LLTEHNICEINGQNIVGLKDSQIA  
 2BYG FQSMTVVEIKLFK.GPKGLGFSIAGGVGNQH.IPGDNSIYVTKIIDGAAQKDGRQVGDRLLMVN  
 3K82 EDIPREPRRIVIHR.GSTGLGFNIVGEGE.....GIFISFILAGGPADLSGELRKGDQILSVNGVDL  
 CASK RSRLVQFQKNTDEPMGIFTLKMNELN.....HCIVARIMHGGMIHROGTLHVGD  
 .TIAM1 GAMGKVTHSIIEKSDTAADTYGFSLSSVED.....GIRRLYVNSVKETGLASKKG.LKAGDEILEIN  
 NRAADALNSSMLKDFLSQP.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	DLG2	L	F	I	V	A	L	L	V	L	A	L	V	V
PSD95	203	205	207	224	233	239	245	248	253	261	265	272	274	276
Cask	L	F	I	I	A	L	I	V	L	A	L	V	I	A
Tiam1	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.

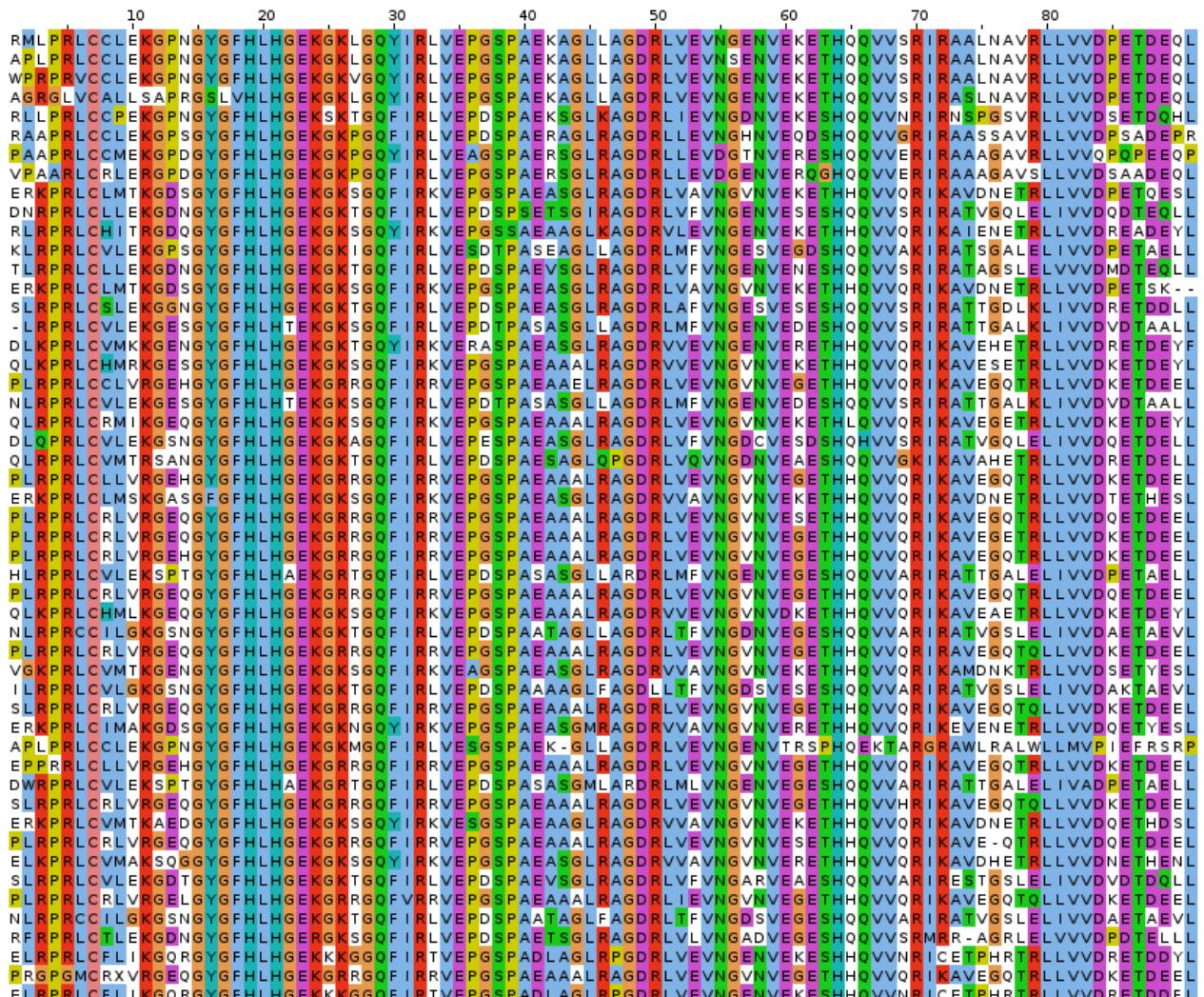


Figure 4.2 – L'alignement de notre sélection de séquences homologues à la protéine NHREF (code PDB :1G9O)

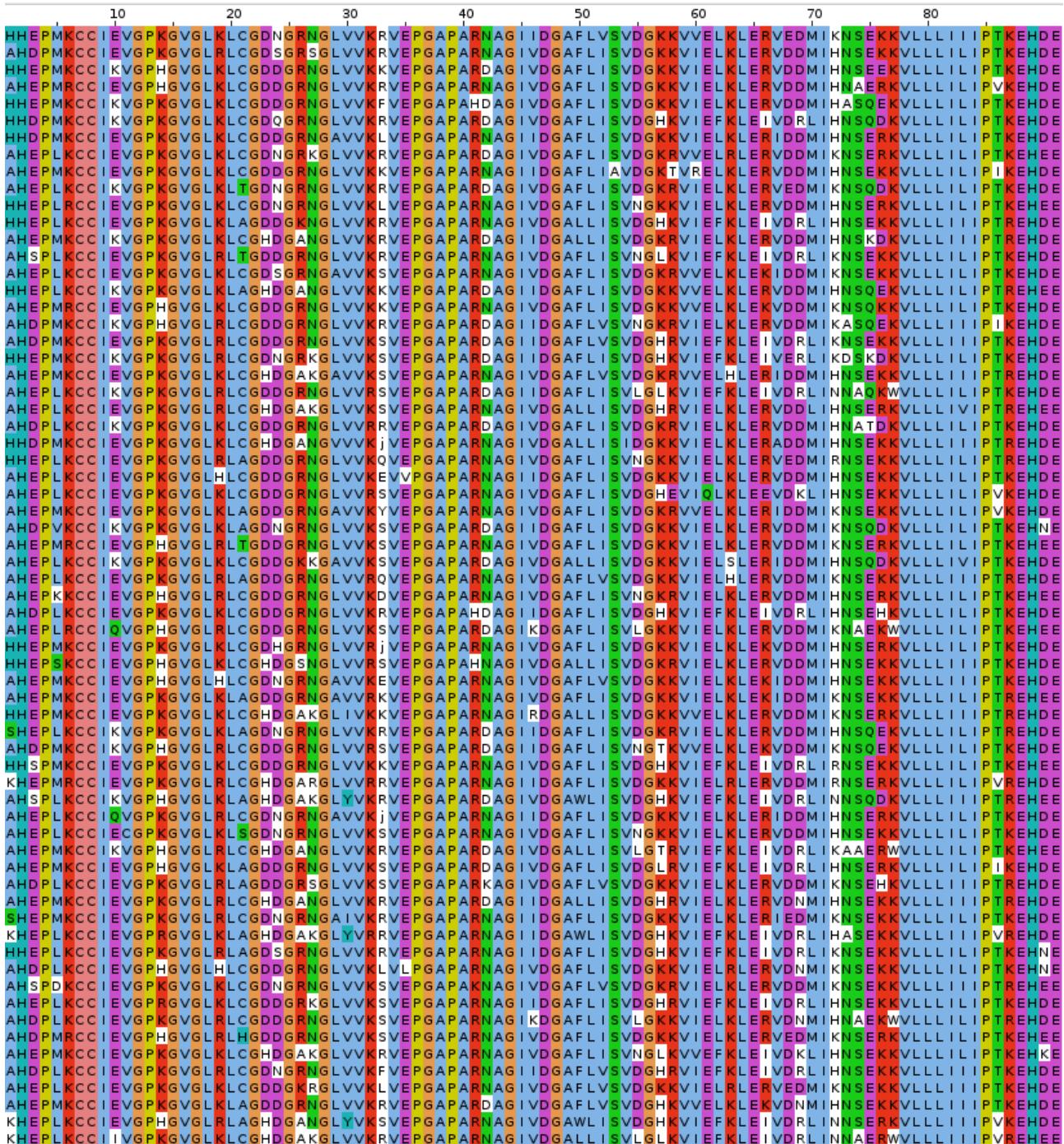


Figure 4.3 – Une sélection de séquences protéiques, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine NHREF (code PDB :1G9O), modèle NEA

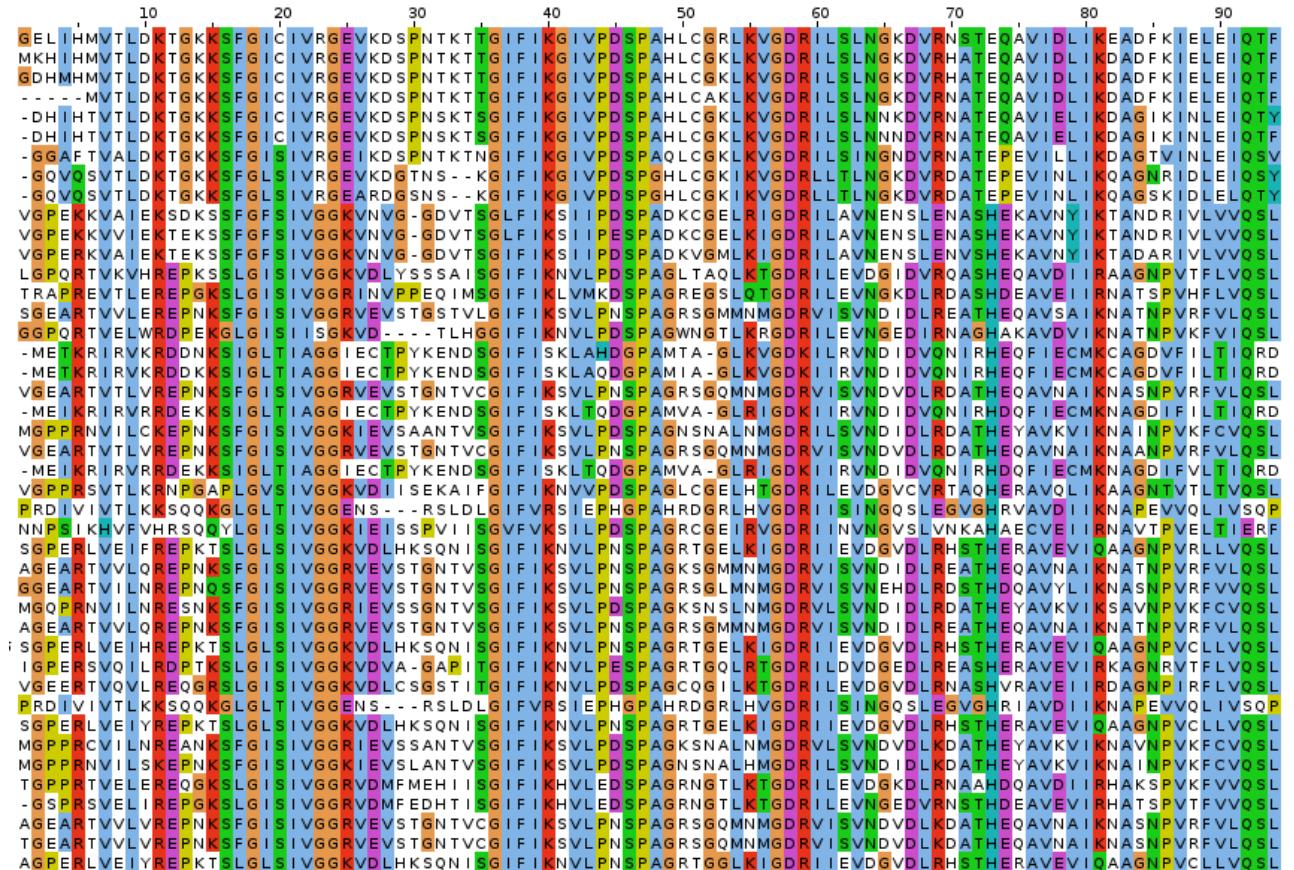


Figure 4.4 – L’alignement de notre sélection de séquences homologues à la protéine INAD (code PDB :1IHJ )

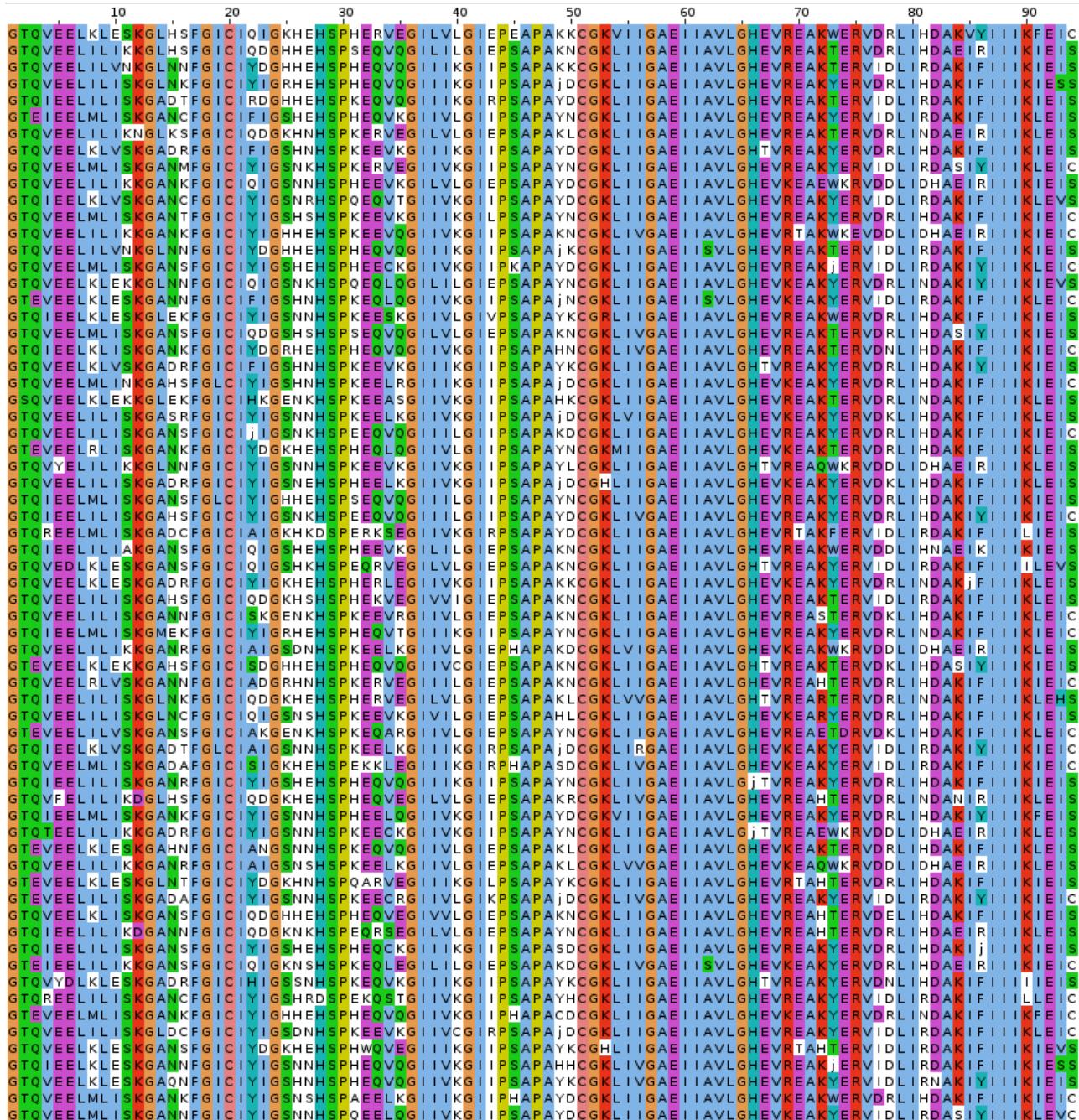


Figure 4.5 – Une sélection de séquences protéins, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine INAD (code PDB :1IHJ), modèle NEA

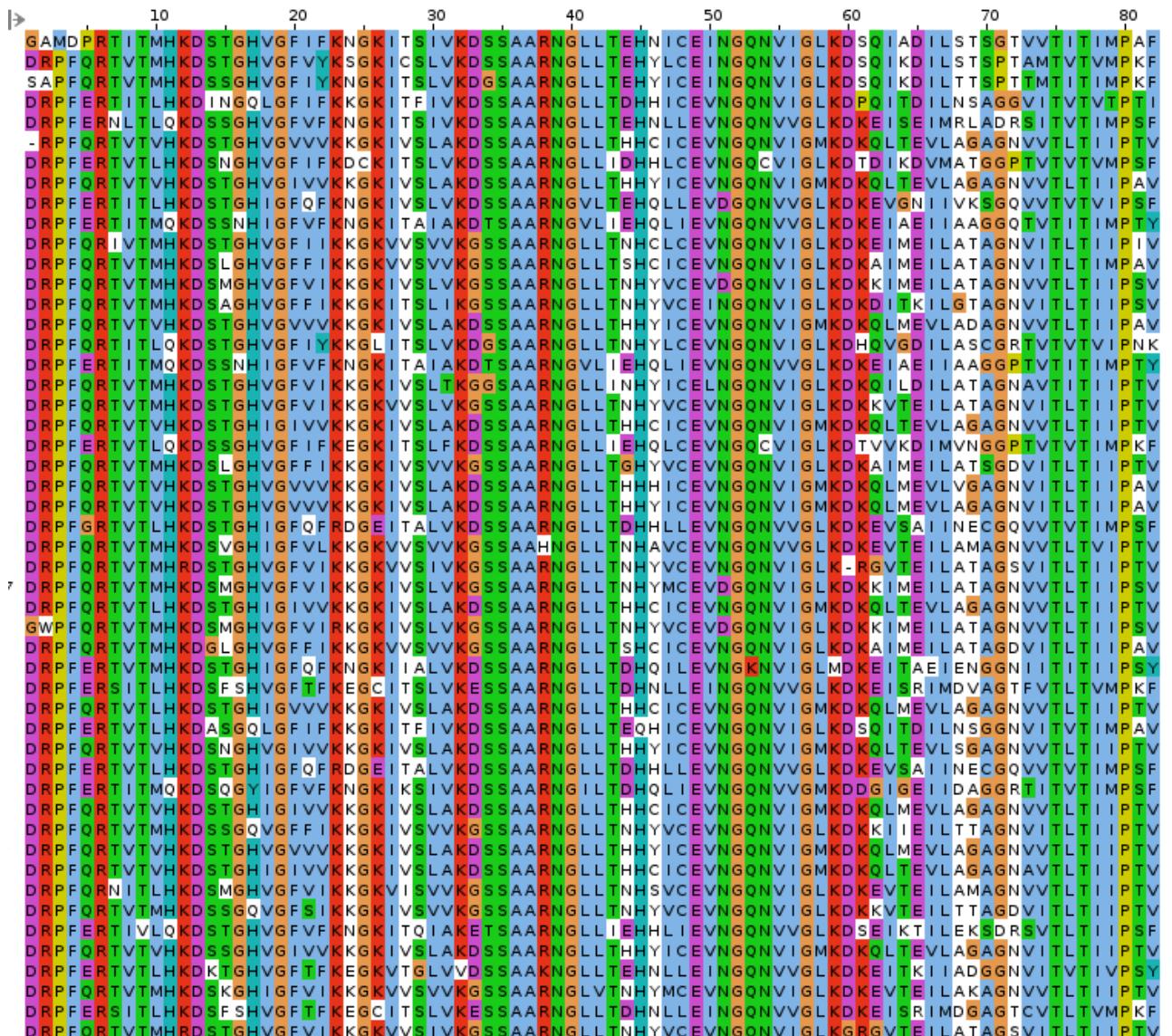


Figure 4.6 – L'alignement de notre sélection de séquences homologues à la protéine Syntenin (code PDB 1R6J)

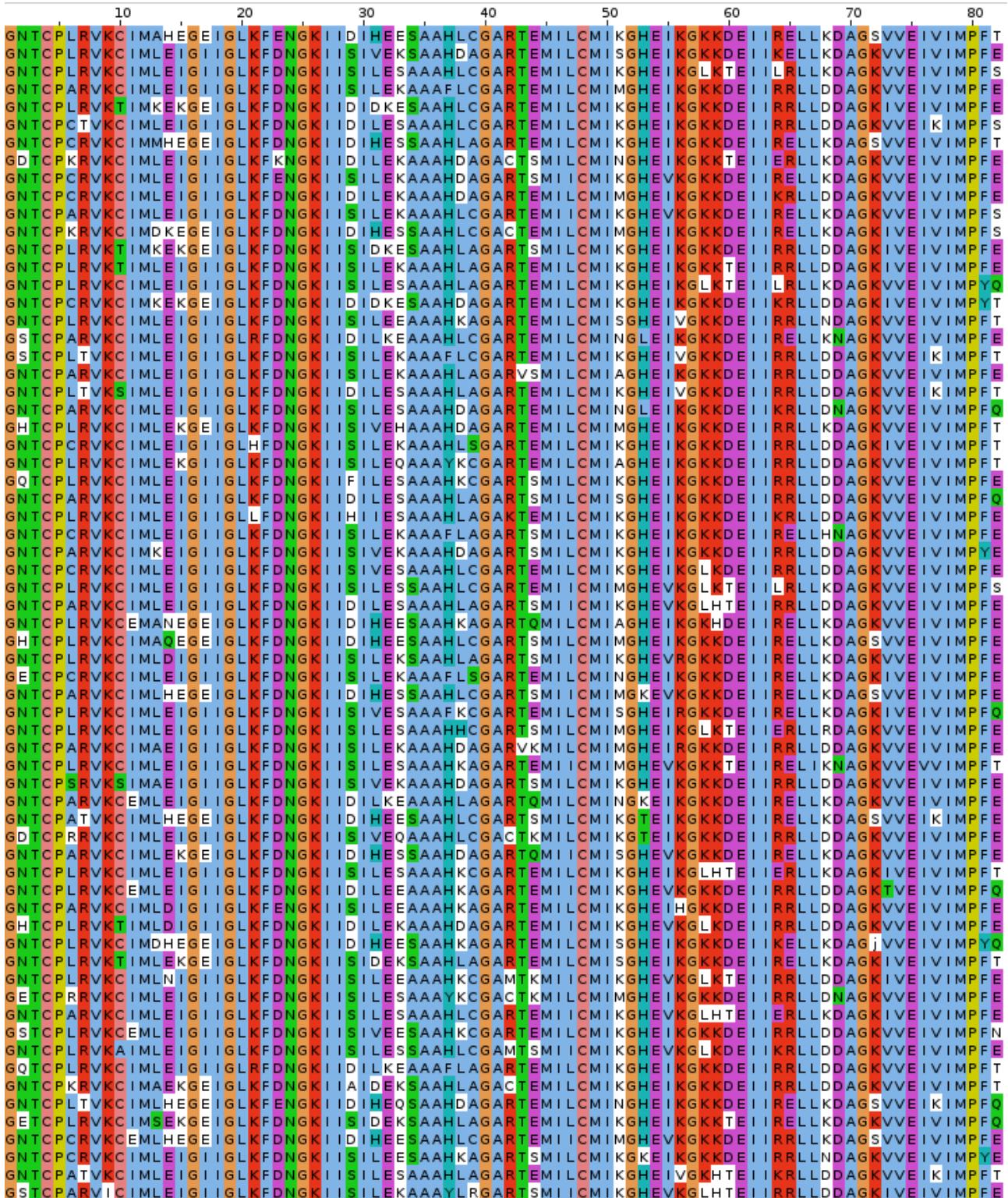


Figure 4.7 – Une sélection de séquences protéiques, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine Syntenin (code PDB :1R6J), modèle NEA

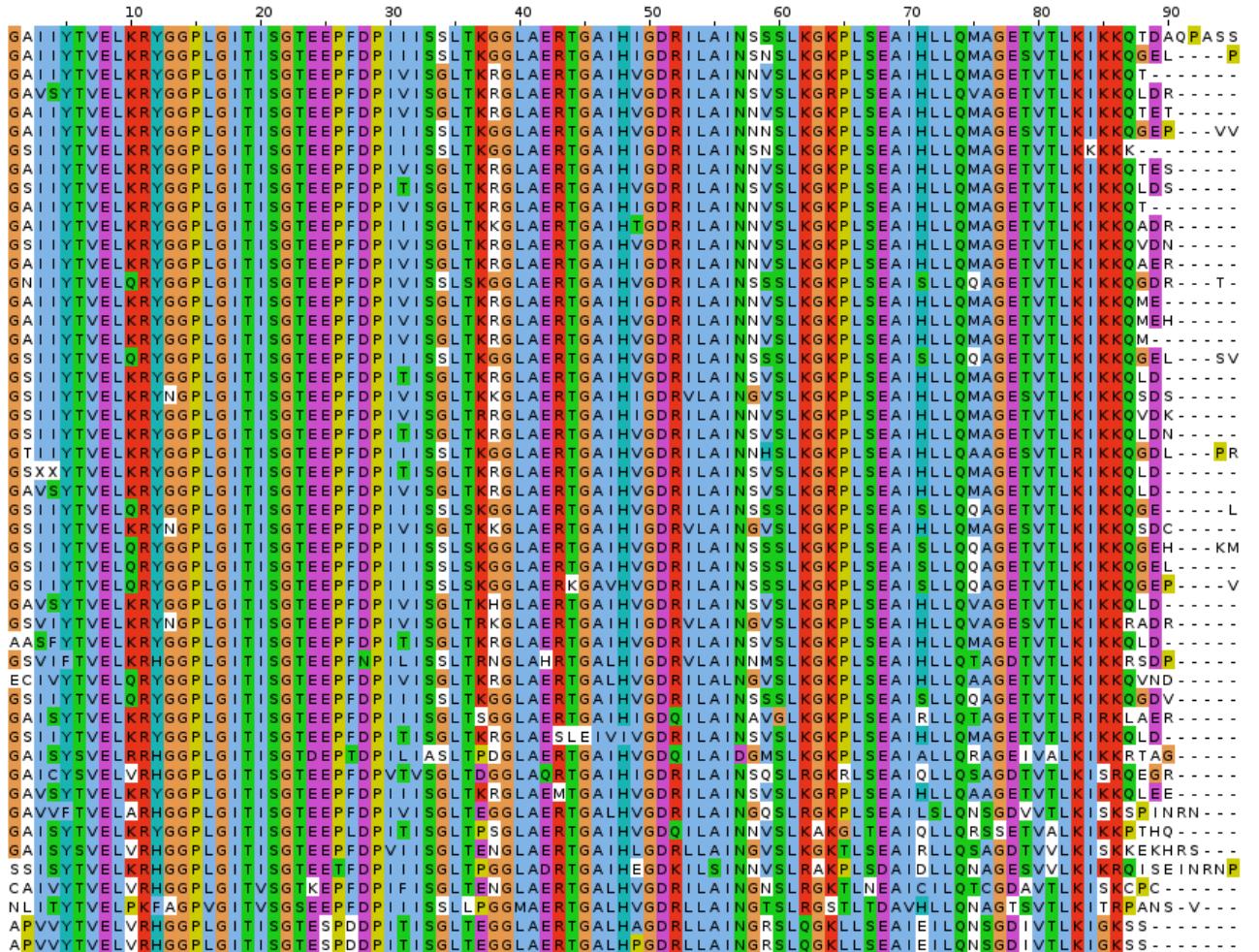


Figure 4.8 – L’alignement de notre sélection de séquences homologues à la protéine GRIP (code PDB : 1N7E)

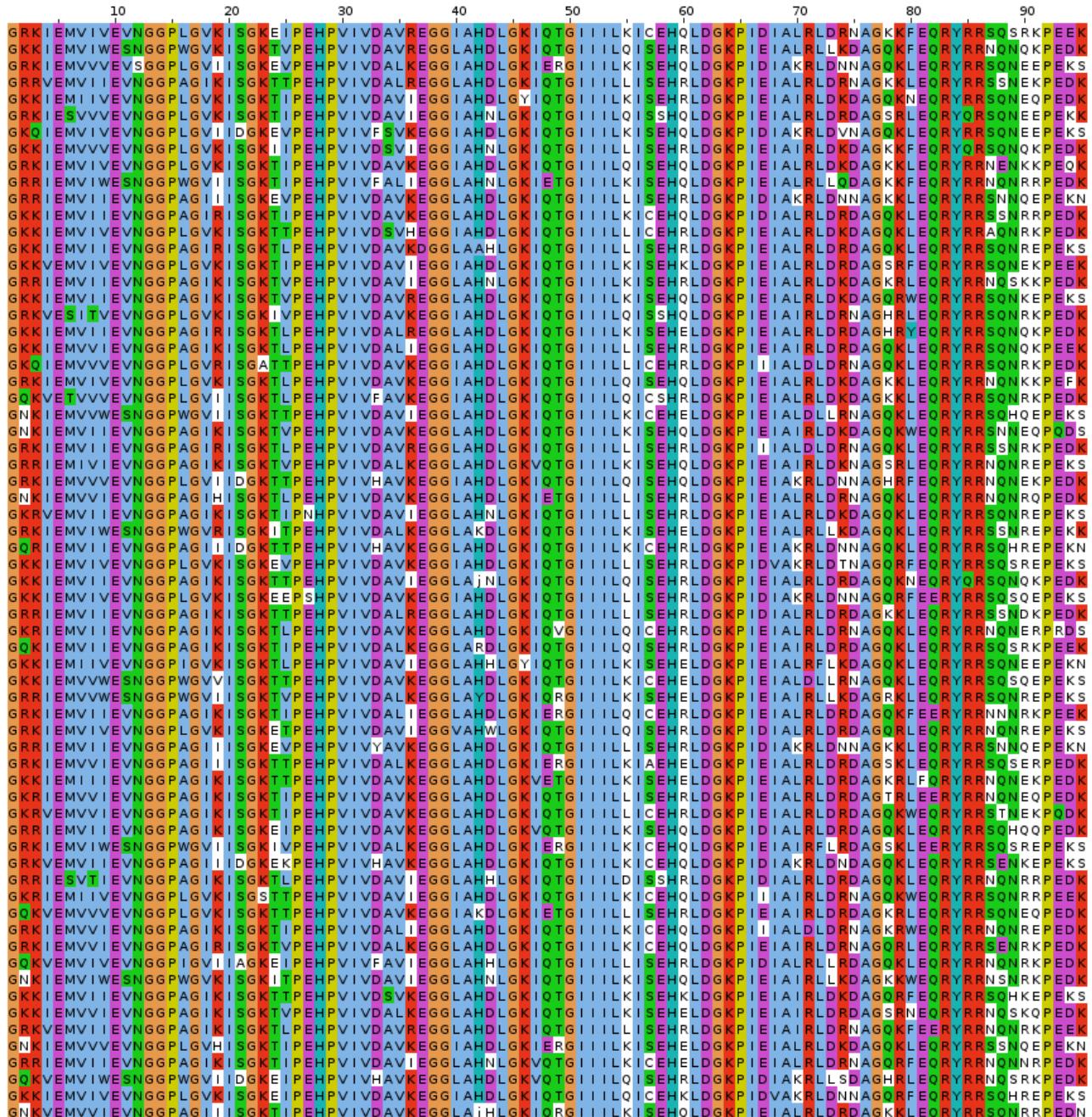


Figure 4.9 – Une sélection de séquences protéins, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine GRIP (code PDB :1N7E), modèle NEA

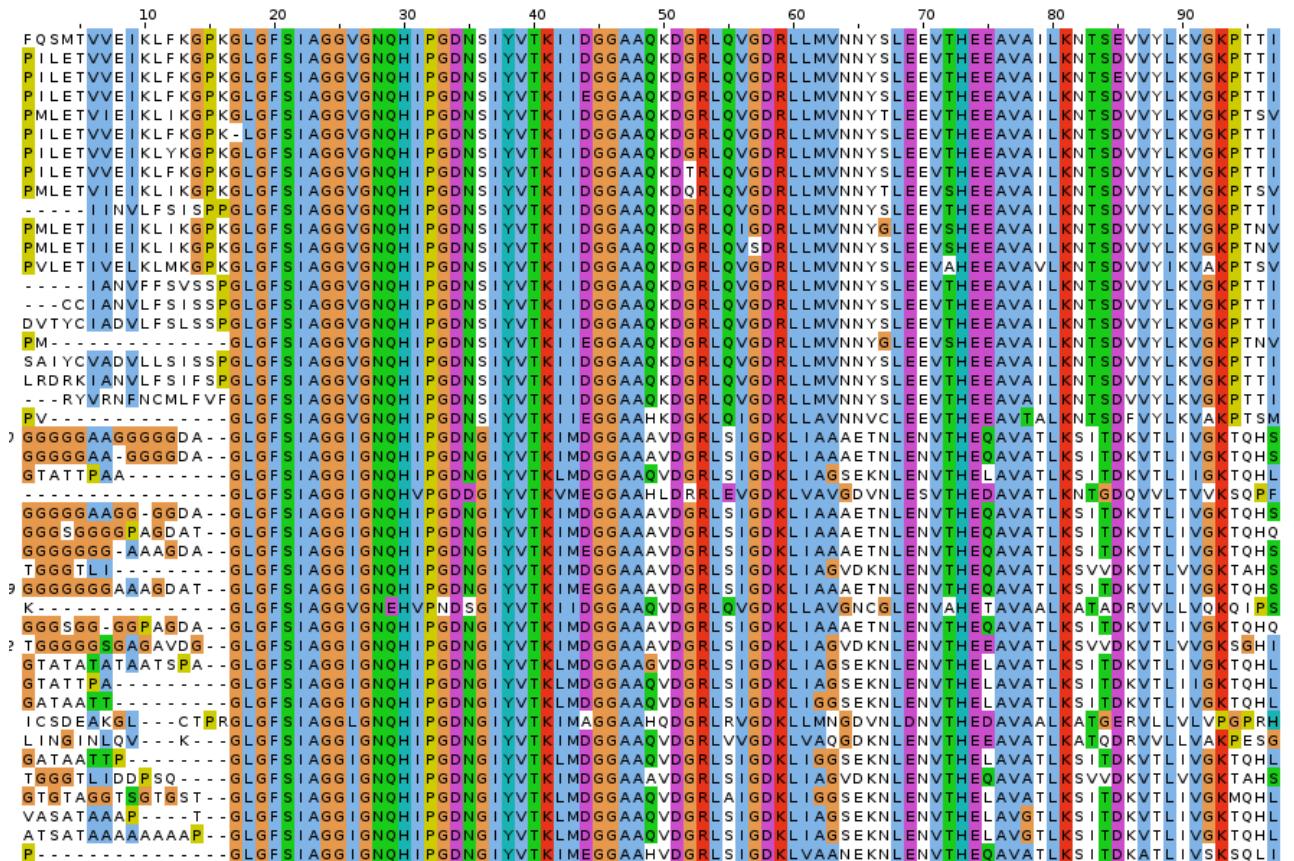


Figure 4.10 – L’alignement de notre sélection de séquences homologues à la protéine DLG2 (code PDB 2BYG)

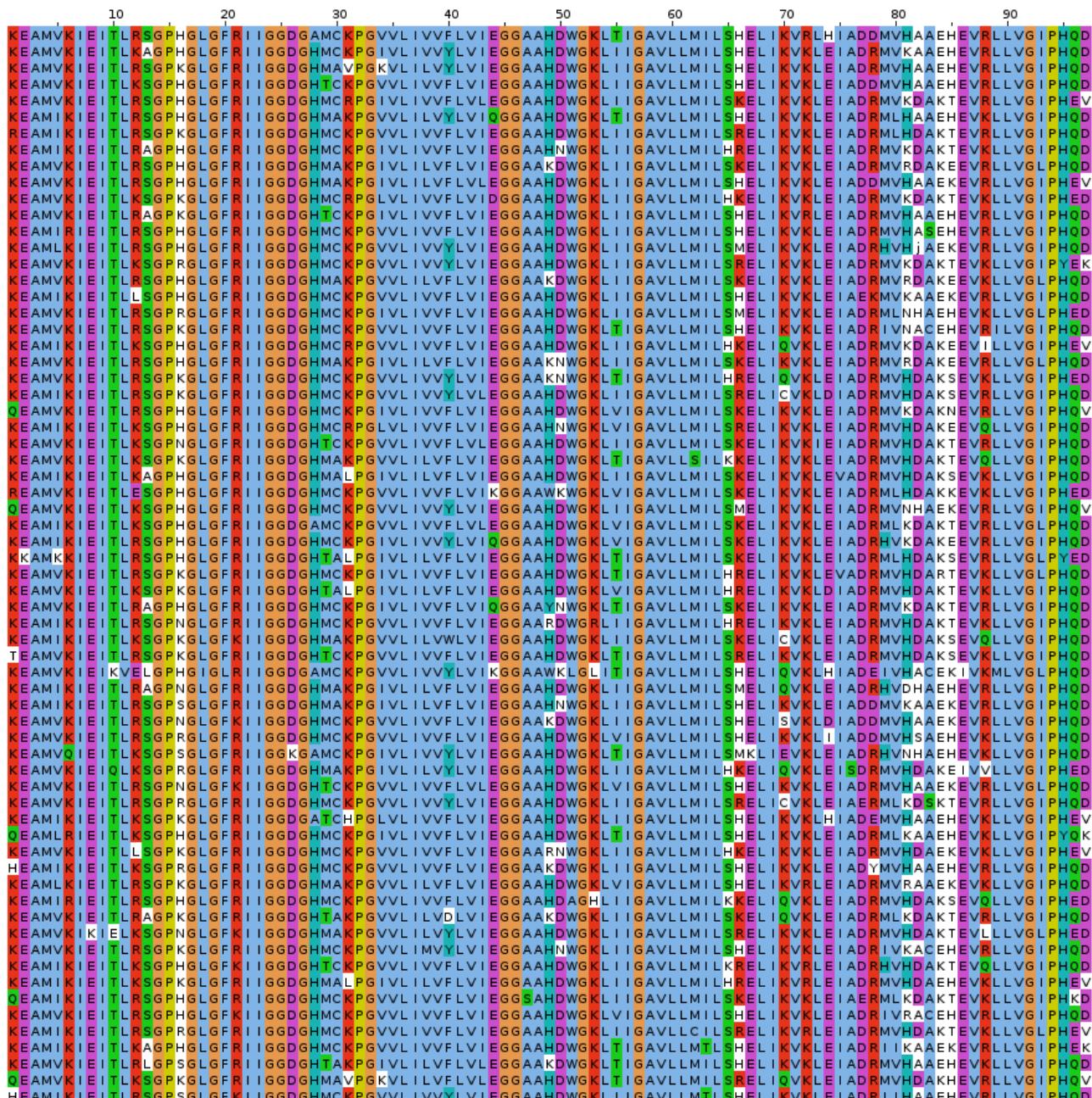


Figure 4.11 – Une sélection de séquences protéiques, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine DLG2 (code PDB 2BYG), modèle NEA

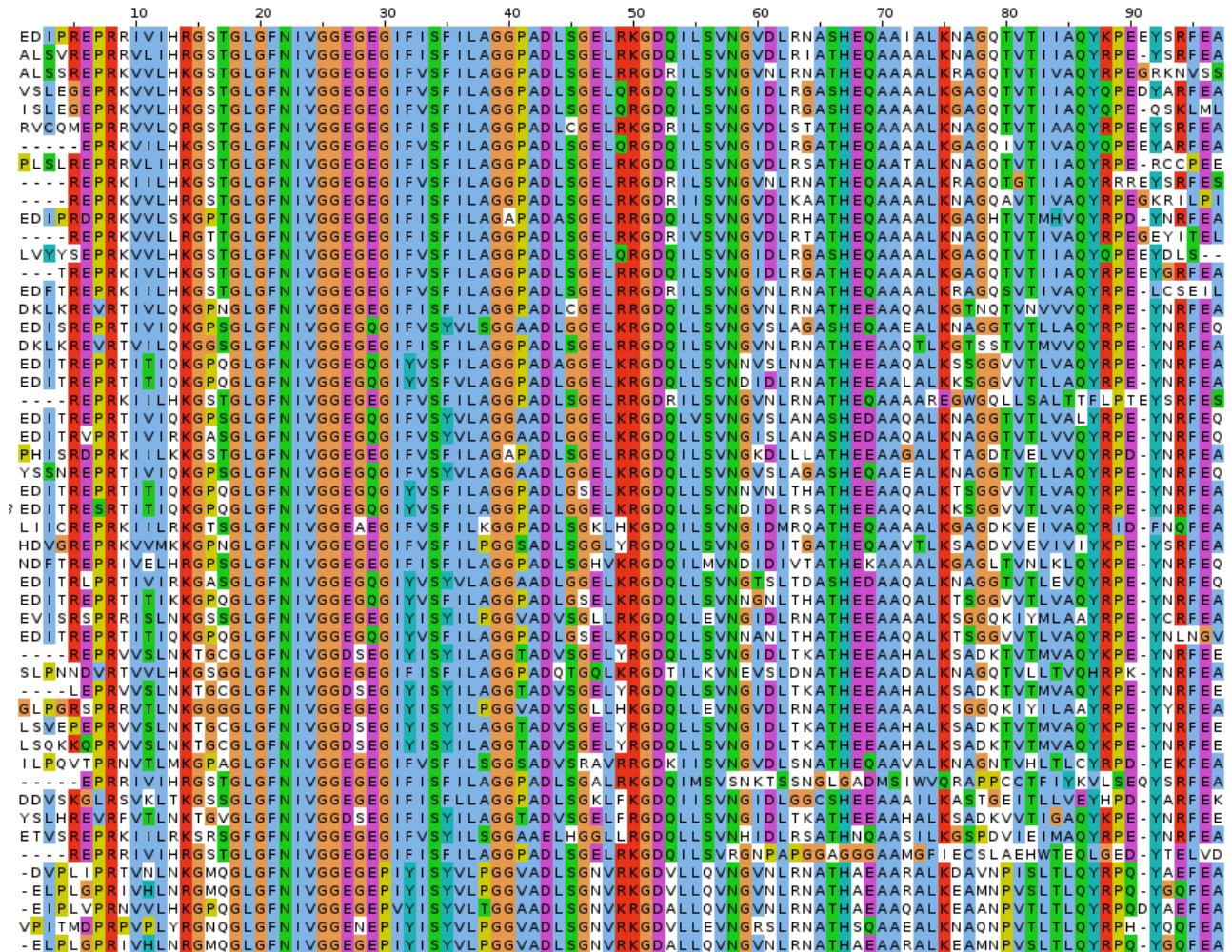


Figure 4.12 – L’alignement de notre sélection de séquences homologues à la protéine PSD95 (code PDB 3K82)

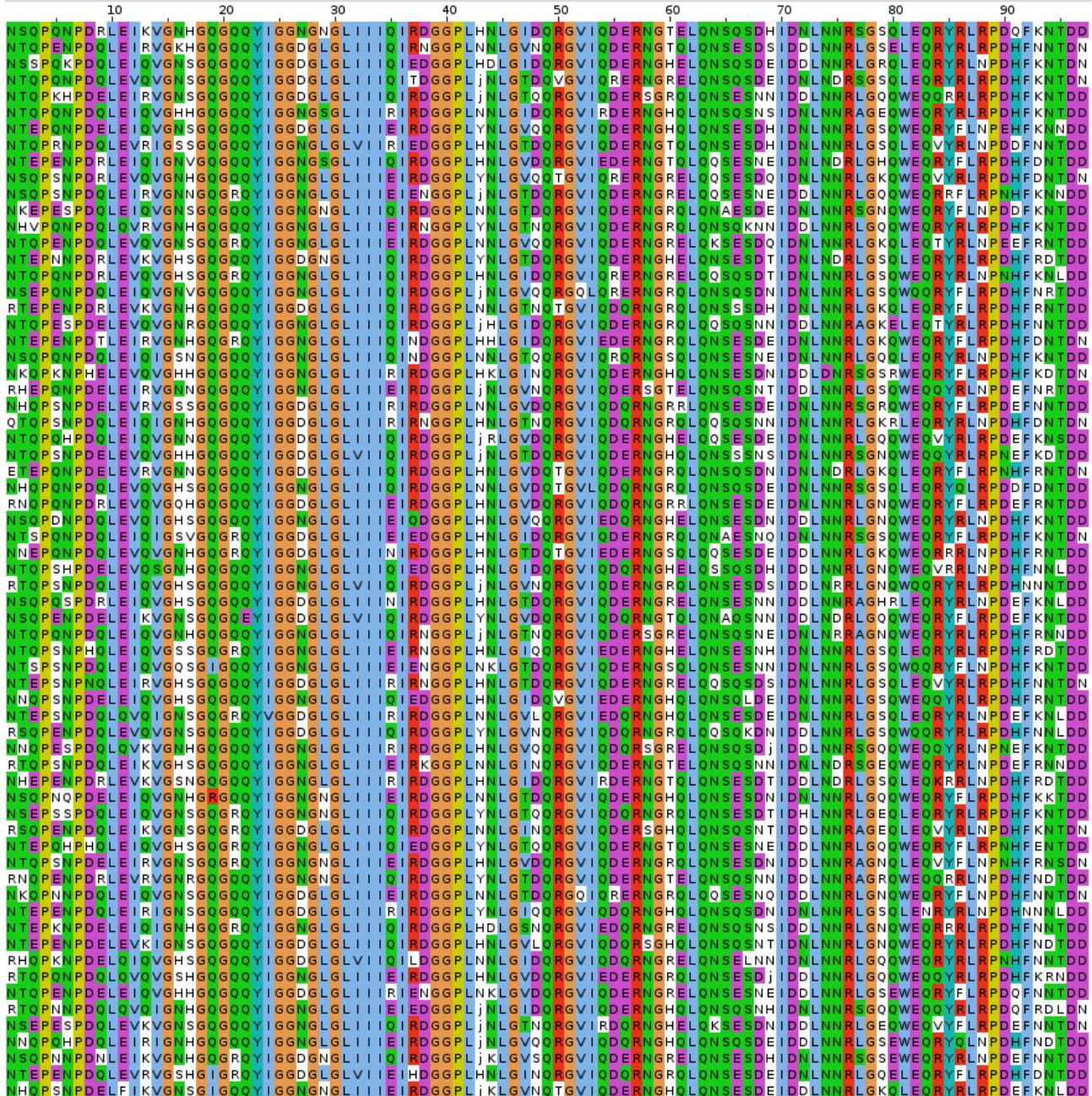


Figure 4.13 – Une sélection de séquences protéins, parmi les 10 000 séquences de meilleure énergie, obtenues avec le backbone de la protéine PSD95 (code PDB :3K82), modèle NEA

## 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 hmmcan 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  $\epsilon_p$  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 enfuies 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		0.0
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  $\delta E_t^r$  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 , 4.15la 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,

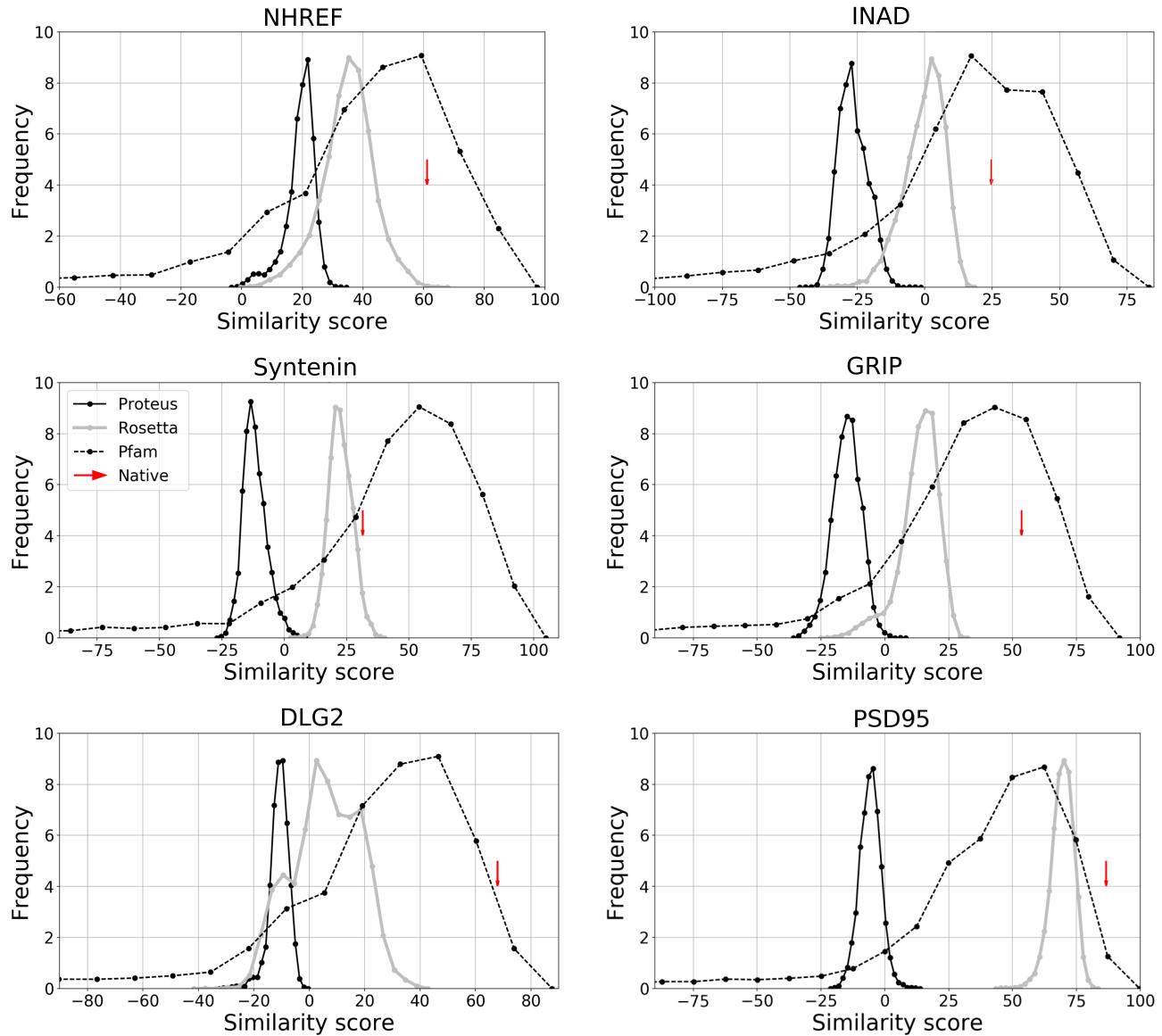


Figure 4.14 – Similarité des séquences des 6 protéines produites par Proteus et Rosetta à l’alignement Pfam RP55, sur l’ensemble des positions.

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.

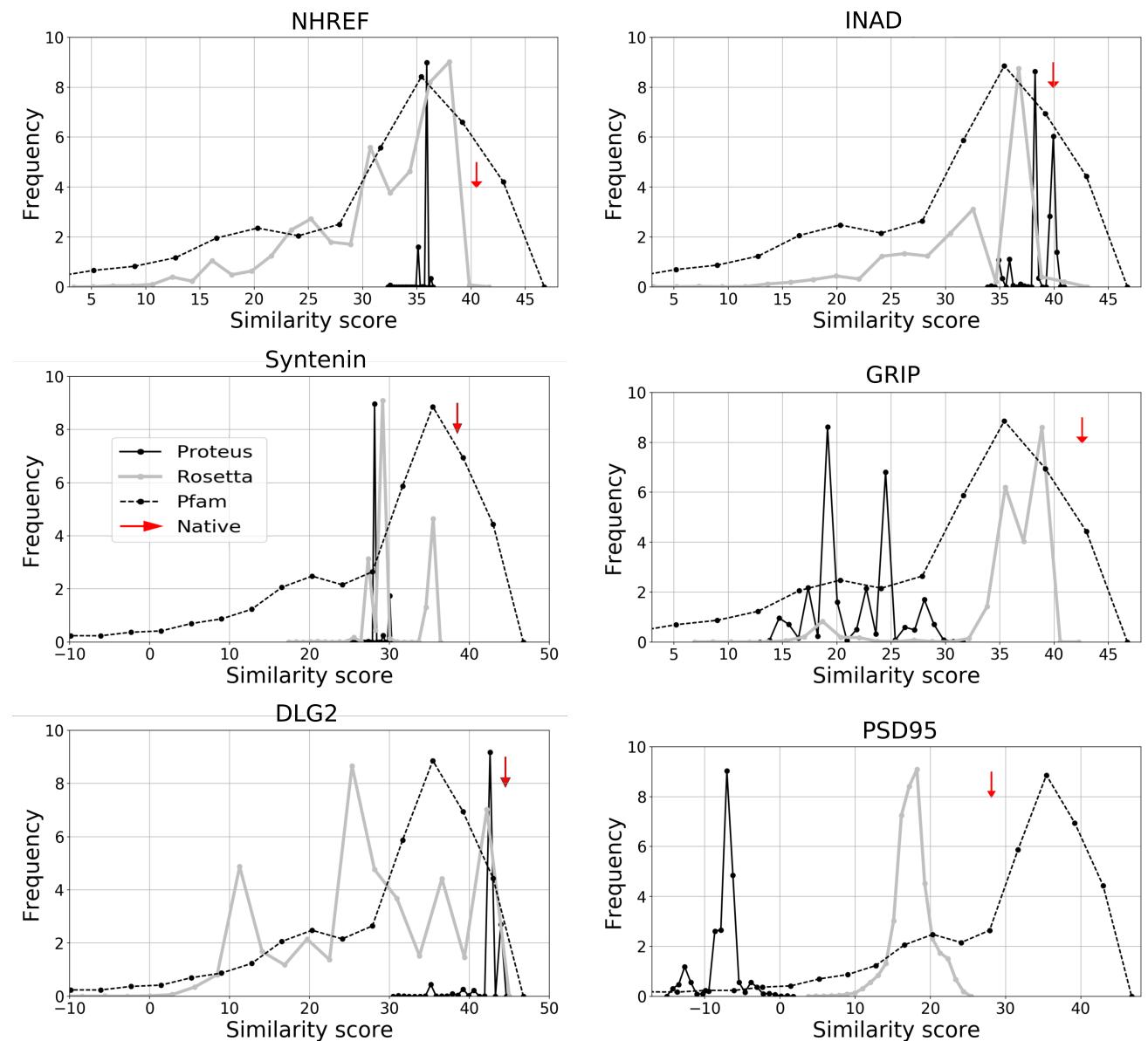


Figure 4.15 – Similarité des séquences des 6 protéines produites par proteus modèle NEA et Rosetta à l’alignement Pfam RP55, sur les positions du cœur hydrophobe.

## 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 <sup>-14</sup>	10000	8.94 10 <sup>-3</sup>	10000
	Syntenin	70/82	2.85 10 <sup>-6</sup>	10000	2.69 10 <sup>-3</sup>	10000
	DLG2	88/97	3.26 10 <sup>-12</sup>	10000	1.96 10 <sup>-3</sup>	10000
Rosetta	NHREF	79/91	1.3 10 <sup>-13</sup>	10000	2.2 10 <sup>-3</sup>	10000
	Syntenin	76/82	7.3 10 <sup>-13</sup>	10000	1.8 10 <sup>-3</sup>	10000
	DLG2	86/97	1,3 10 <sup>-9</sup>	10 000	9,6 10 <sup>-4</sup>	10 000
Proteus NEA epsilon=4	NHREF	62/91	3,22 10 <sup>-3</sup>	9857	1,00 10 <sup>-2</sup>	9857
	Syntenin	70/82	2.83 10 <sup>-3</sup>	9879	3,62 10 <sup>-3</sup>	9879
	DLG2	83/97	1,66 10 <sup>-3</sup>	9876	3,18 10 <sup>-3</sup>	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

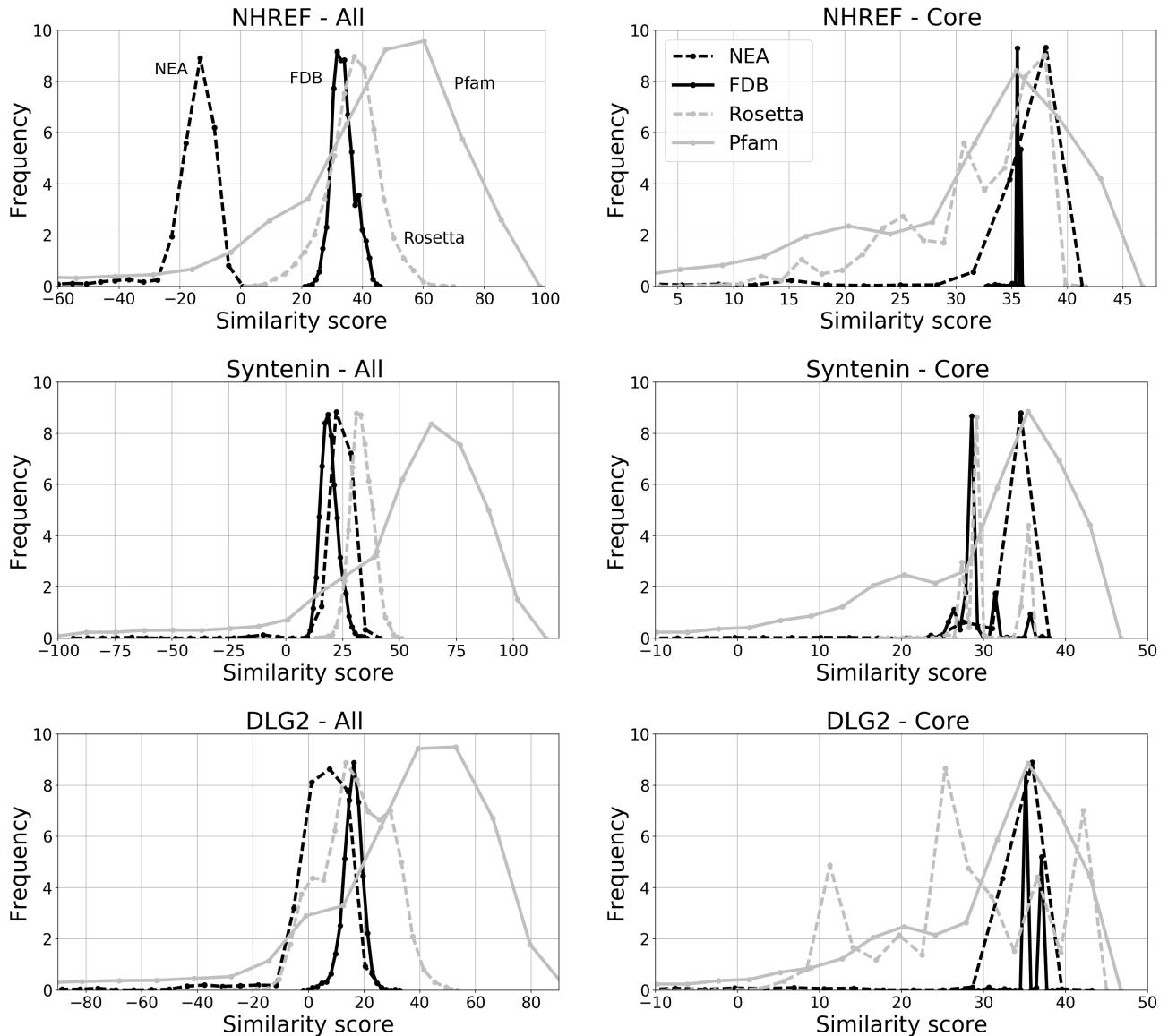


Figure 4.16 – Similarité des séquences Proteus (NEA et FDB), Rosetta , et des séquences de l’alignement Pfam RP55, sur toutes les positions à gauche et sur les positions du cœur à droite.

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

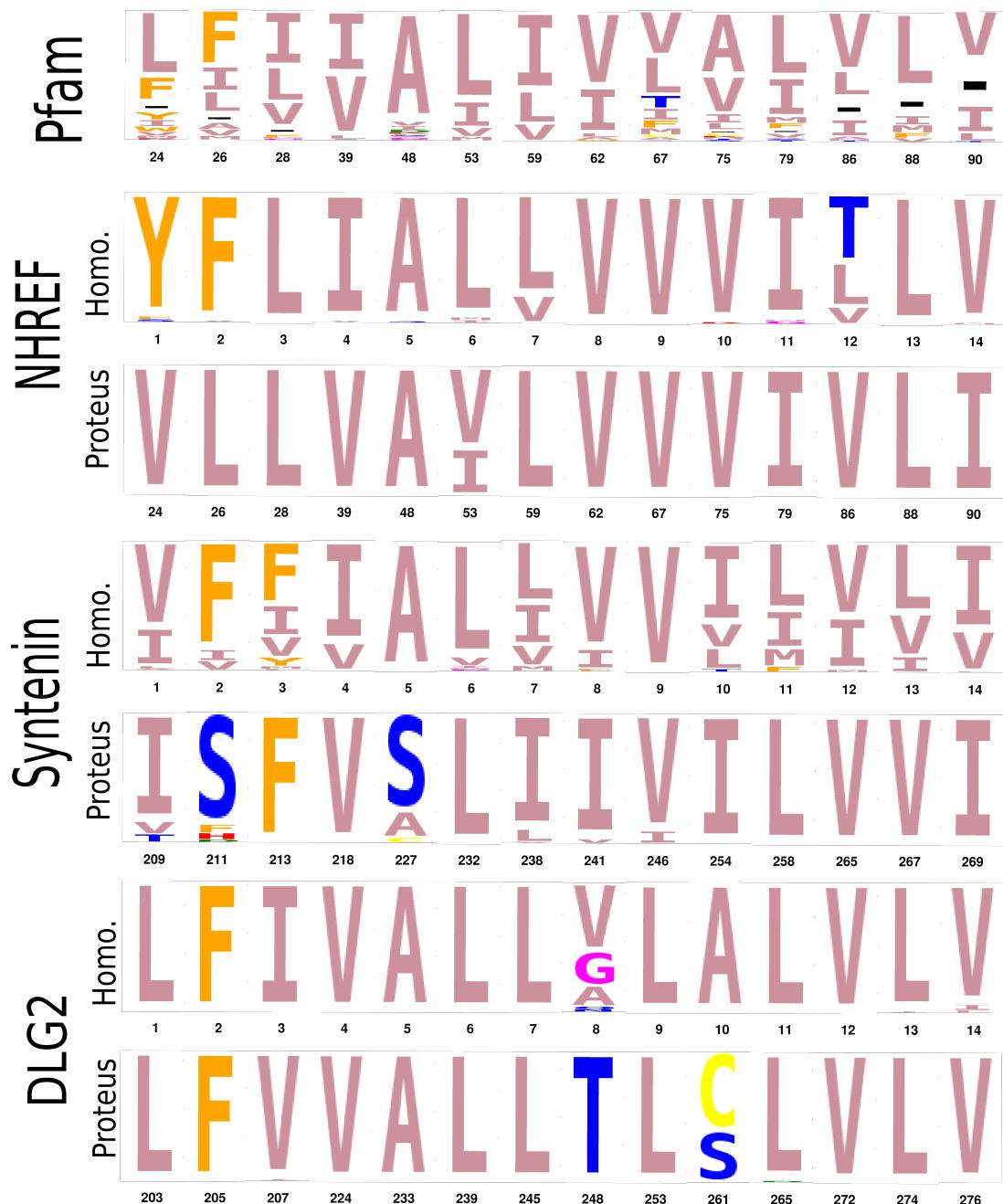


Figure 4.17 – Les séquences conçues par Proteus et les séquences naturelles représentées sous forme de logo pour les positions du cœur hydrophobe

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

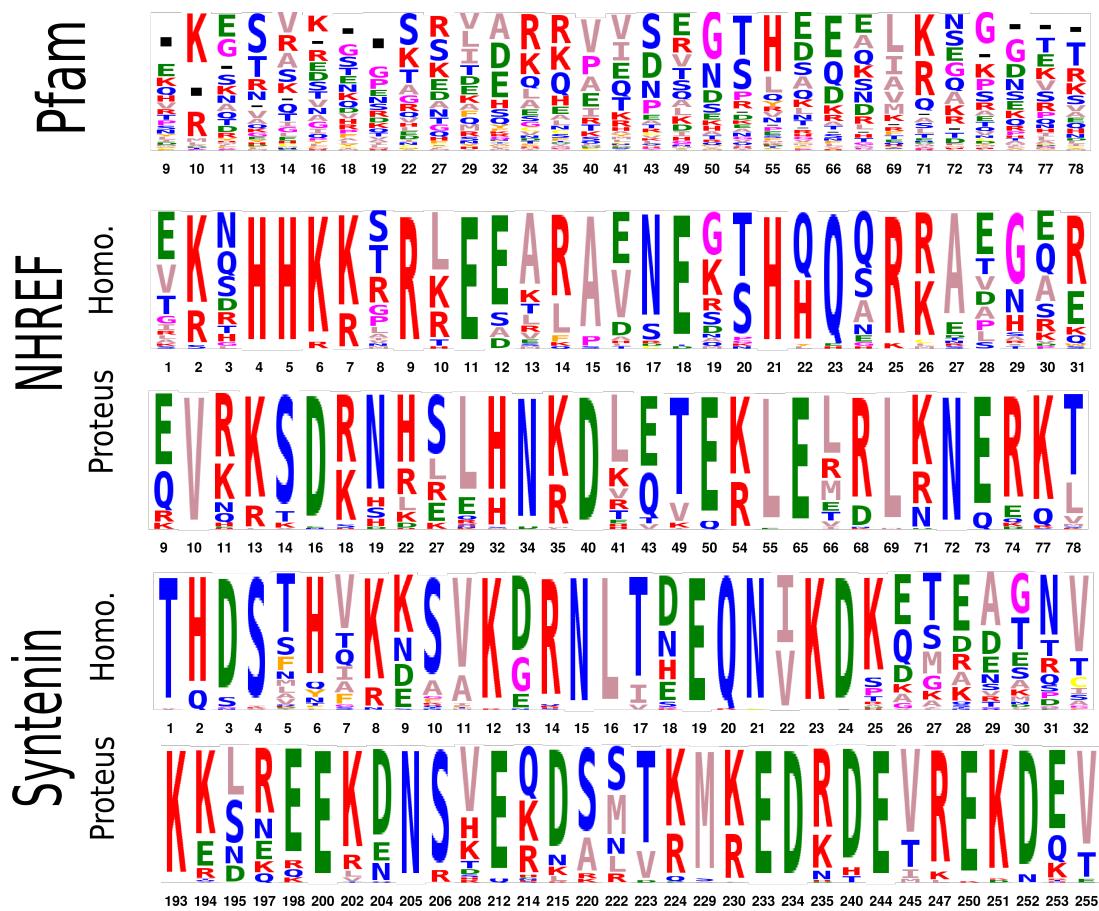
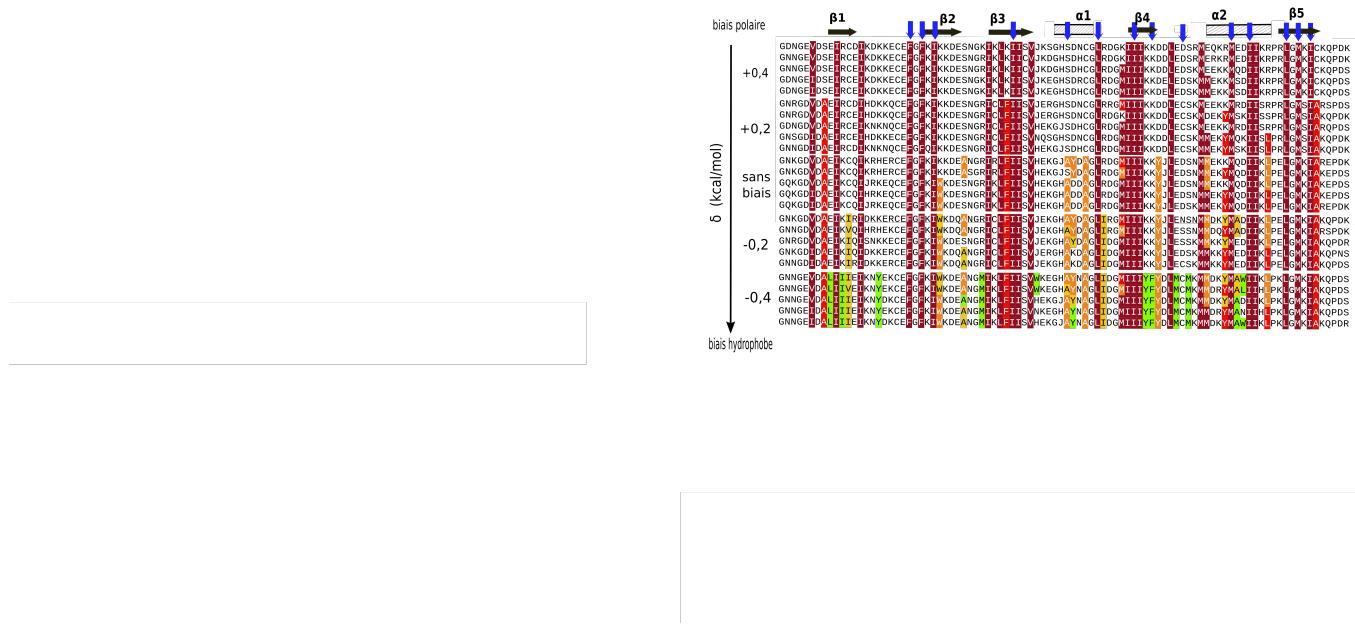


Figure 4.18 – Les séquences conçues par Proteus et les séquences naturelles représentées sous forme de logo pour les positions du cœur hydrophobe

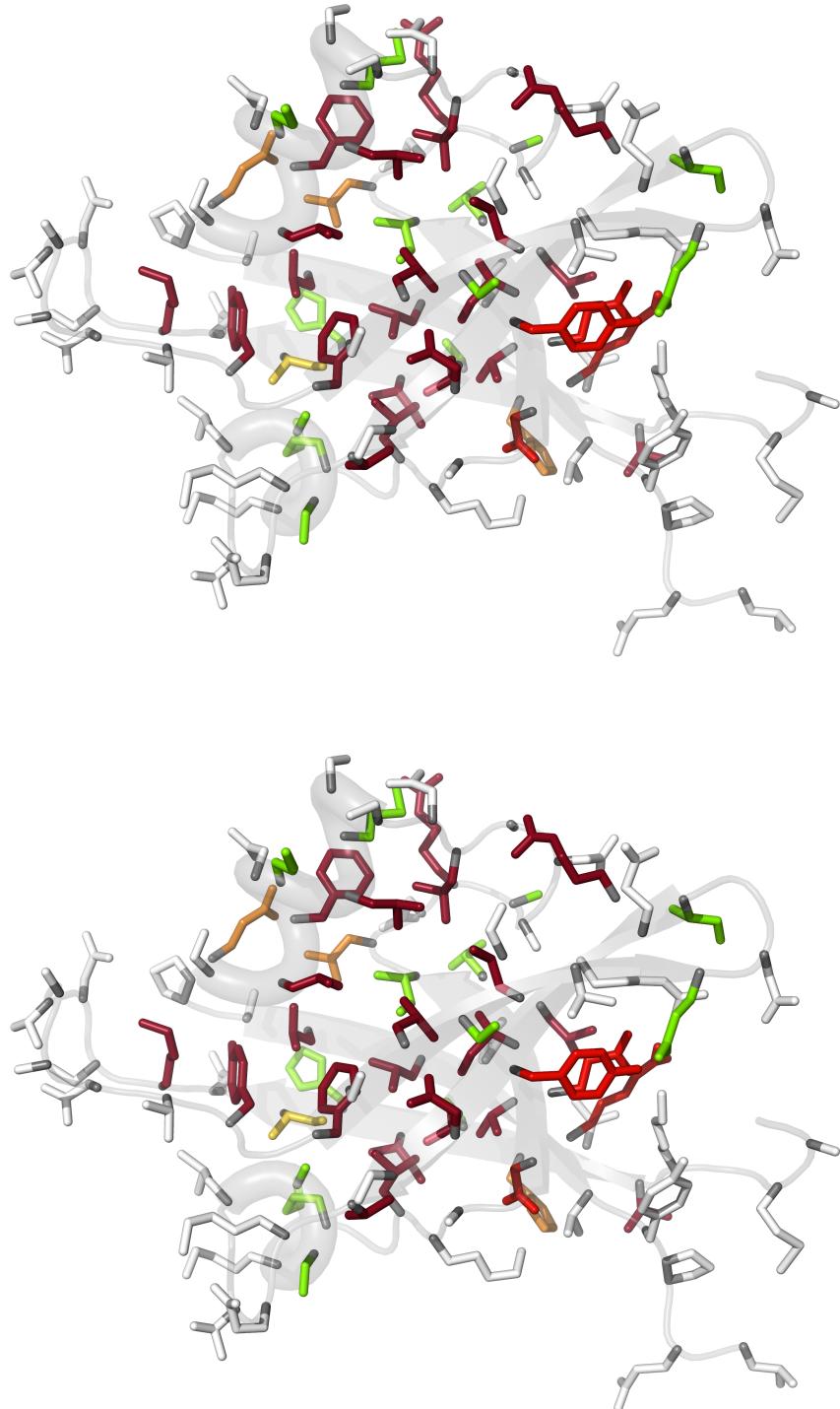
comprend un terme d'énergie de biais  $\delta = -0.4 \text{ kcal/mol}$  (par position) qui favorise les types hydrophobes. En diminuant progressivement la valeur du biais d'énergie  $\delta$ , nous « titrons » efficacement les résidus hydrophobes.

#### 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  $\delta$  fixé parmi les séquences de plus faible énergie.

Figure 4.19 – Structure native Tiam1 avec les hydrophobes pour des  $\delta$  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.



#### **4.9. Application : Croissance du noyau hydrophobe**

---

Les résultats pour Tiam1 sont présentés à la figure ?? et à la figure 4.19. À la plus grande valeur de  $\delta$  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  $\delta$  à  $0,2\text{kcal/mol}$ , le noyau hydrophobe ne compte plus que 4 ou 5 changements en type polaire. À la valeur de  $\delta$  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  $\delta$  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  $\psi_h$  est défini comme le nombre  $\delta N$  de positions qui ont changées de non polaire à polaire, divisé par le produit de la variation  $\delta E$  dans l'énergie de polarisation et le nombre moyen  $N$  de position non polaire à biais nul. Nous appelons  $\psi_h$  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.

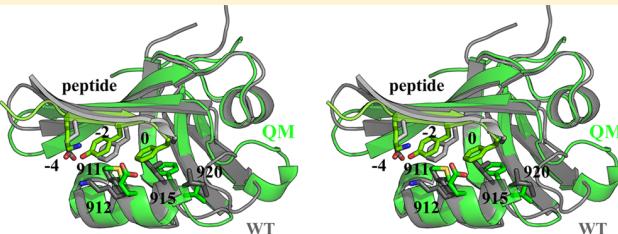
# Computational Design of the Tiam1 PDZ Domain and Its Ligand Binding

David Mignon,<sup>§,†</sup> Nicolas Panel,<sup>§,†,ID</sup> Xingyu Chen,<sup>†</sup> Ernesto J. Fuentes,<sup>‡</sup> and Thomas Simonson<sup>\*,†,ID</sup>

<sup>†</sup>Laboratoire de Biochimie (CNRS UMR7654), Ecole Polytechnique, Palaiseau, France

<sup>‡</sup>Department of Biochemistry, Roy J. & Lucille A. Carver College of Medicine and Holden Comprehensive Cancer Center, University of Iowa, Iowa City, Iowa 52242-1109, United States

<sup>§</sup> Supporting Information



**ABSTRACT:** PDZ domains direct protein–protein interactions and serve as models for protein design. Here, we optimized a protein design energy function for the Tiam1 and Cask PDZ domains that combines a molecular mechanics energy, Generalized Born solvent, and an empirical unfolded state model. Designed sequences were recognized as PDZ domains by the Superfamily fold recognition tool and had similarity scores comparable to natural PDZ sequences. The optimized model was used to redesign the two PDZ domains, by gradually varying the chemical potential of hydrophobic amino acids; the tendency of each position to lose or gain a hydrophobic character represents a novel hydrophobicity index. We also redesigned four positions in the Tiam1 PDZ domain involved in peptide binding specificity. The calculated affinity differences between designed variants reproduced experimental data and suggest substitutions with altered specificities.

## 1. INTRODUCTION

PDZ domains (“Postsynaptic density-95/Discs large/Zonula occludens-1”) are small, globular protein domains that establish protein–protein interaction networks in the cell.<sup>1–6</sup> They form specific interactions with other, target proteins, usually by recognizing a few amino acids at the target C-terminus. Because of their biological importance, PDZ domains and their interaction with target proteins have been extensively studied and computationally engineered. Peptide ligands have been designed that modulate the activity of PDZ domains involved in various pathologies.<sup>7–9</sup> Engineered PDZ domains and PDZ ligands have been used to elucidate principles of protein folding and evolution.<sup>10–13</sup> In addition, these small domains with their peptide ligands provide benchmarks to test the computational methods themselves.<sup>14–16</sup>

An emerging method that has been applied to several PDZ domains is computational protein design (CPD).<sup>17–22</sup> Starting from a three-dimensional (3D) structural model, CPD explores a large space of amino acid sequences and conformations to identify protein variants that have predefined properties, such as stability or ligand binding. Conformational space is usually defined by a discrete or continuous library of side chain rotamers and by a finite set of backbone conformations or a specific repertoire of allowed backbone deformations. The energy function that drives CPD usually combines physical and

empirical terms,<sup>23–25</sup> while the solvent and the protein unfolded state are described implicitly.<sup>41</sup>

Here, we considered a simple but important class of CPD models. The energy is a physics-based function of the “MMGBSA” type, which combines a molecular mechanics protein energy with a Generalized Born + surface area implicit solvent. The folded protein is represented by a single, fixed, backbone conformation and a discrete side chain rotamer library. The unfolded state energy depends only on sequence composition, not an explicit structural model. The main adjustable model parameters are the protein dielectric constant  $\epsilon_p$ , a small set of atomic surface energy coefficients  $\sigma_b$ , and a collection of amino acid chemical potentials, or “reference energies”  $E_t^r$ . Each surface coefficient measures the preference of a particular atom type to be solvent-exposed, while each reference energy represents the contribution of a single amino acid of type  $t$  to the unfolded state energy. The model is implemented in the Proteus software.<sup>26–28</sup>

The present physics-based energy function can be compared to more empirical ones, of which the most successful is the Rosetta energy function.<sup>29–31</sup> The Rosetta function includes a Lennard-Jones repulsion term, a Coulomb term, a hydrogen-bonding term, a Lazaridis–Karplus solvation term,<sup>32</sup> and

**Received:** December 28, 2016

**Published:** April 10, 2017

63 unfolded state reference energies. It has a large number of  
 64 parameters specifically optimized for CPD, which provide  
 65 optimal performance, but less transferability and a less  
 66 transparent physical interpretation. Proteus also provides  
 67 some specific functionalities, such as Replica Exchange Monte  
 68 Carlo, various importance sampling methods, and the ability to  
 69 compute free energies that are formally exact.<sup>33–35</sup>

70 We optimized the reference energies  $E_t^r$  for the Tiam1 and  
 71 Cask PDZ proteins, using a maximum likelihood formalism. We  
 72 compared two values of the protein dielectric constant,  $\epsilon_p = 4$   
 73 and 8. These values gave good results in a systematic study that  
 74 compared dielectric constants in the range 1–32.<sup>36</sup> The  
 75 performance of the model was tested by generating designed  
 76 sequences for both proteins and comparing them to natural  
 77 sequences, as well as sequences generated with the Rosetta  
 78 energy function and software.<sup>37</sup> The sequence design was  
 79 performed by running long Monte Carlo simulations in which  
 80 all protein positions except Gly and Pro were allowed to mutate  
 81 freely, leading to thousands of designed protein variants. The  
 82 testing included cross-validation, for which the reference  
 83 energies were optimized using one set of PDZ domains, then  
 84 applied to others. We also performed 100–1000 ns molecular  
 85 dynamics (MD) simulations for a few of the sequences  
 86 designed with our optimized CPD model, to help assess their  
 87 stability. Ten sequences were stable over 100 ns or more and  
 88 one over 1000 ns of MD simulation.

89 We then applied the CPD model with optimized parameters  
 90 to two problems, which are representative of the two main  
 91 areas we are interested in exploring: the plasticity of sequence  
 92 space for PDZ domains and designing strong and specific PDZ  
 93 ligands. Earlier applications in these areas mostly employed  
 94 empirical, knowledge-based energy functions such as the  
 95 Rosetta function.<sup>9,10,13,14</sup> First, we performed a series of  
 96 Monte Carlo simulations of two PDZ domains where the  
 97 chemical potential of the hydrophobic amino acid types was  
 98 gradually increased, artificially biasing the protein composition.  
 99 As the hydrophobic bias was increased, hydrophobic amino  
 100 acids gradually invaded the protein from the inside out, forming  
 101 a hydrophobic core that became larger than the natural one.  
 102 The propensity of each core position to become hydrophobic at  
 103 a high or low level of bias can be seen as a structure-dependent  
 104 hydrophobicity index, which provides information on the  
 105 designability or plasticity of the protein core. The second  
 106 application consisted in designing four Tiam1 positions known  
 107 to be involved in specific target recognition. These four  
 108 positions were varied through Monte Carlo simulations of  
 109 either the apoprotein or the protein in complex with two  
 110 distinct peptide ligands. The simulations were in agreement  
 111 with experimental sequences and binding affinities, and suggest  
 112 new variants that could have altered specificities. This  
 113 application is a step toward the design of strong peptide  
 114 binders, which could be of use as reagents or inhibitors *in vitro*  
 115 or *in vivo*.

## 2. THE UNFOLDED STATE MODEL

116 **2.1. Maximum Likelihood Reference Energies.** The  
 117 Monte Carlo method employed here generates a Markov chain  
 118 of states,<sup>38,39</sup> such that the states are populated according to a  
 119 Boltzmann distribution. The energy employed is not the folded  
 120 protein's energy, but rather its *folding* energy, that is, the  
 121 difference between its folded and unfolded state energies.<sup>33</sup>  
 122 One possible elementary move is a “mutation”, we modify the  
 123 side chain type  $t \rightarrow t'$  at a chosen position  $i$  in the folded

protein, assigning a particular rotamer  $r'$  to the new side chain.<sup>124</sup>  
 We consider the same mutation in the unfolded state. For a  
 125 particular sequence  $S$ , the unfolded state energy has the form:<sup>126</sup>

$$E^u = \sum_{i \in S} E^r(t_i) \quad (1) \quad 127$$

The sum is over all amino acids;  $t_i$  represents the side chain  
 128 type at position  $i$ . The type-dependent quantities  $E^r(t) \equiv E_t^r$  are  
 129 referred to as “reference energies”; they can be thought of as  
 130 effective chemical potentials of each amino acid type. The  
 131 energy change due to a mutation has the form:<sup>132</sup>

$$\begin{aligned} \Delta E &= \Delta E^f - \Delta E^u \\ &= (E^f(\dots t'_i, r'_i \dots) - E^f(\dots t_i, r_i \dots)) - (E^r(t'_i) - E^r(t_i)) \end{aligned} \quad (2) \quad 133$$

where  $\Delta E^f$  and  $\Delta E^u$  are the energy changes in the folded and  
 134 unfolded state, respectively. The reference energies are essential  
 135 parameters in the simulation model. Our goal here is to choose  
 136 them empirically so that the simulation produces amino acid  
 137 frequencies that match a set of target values, for example  
 138 experimental values in the Pfam database. Specifically, we will  
 139 choose them so as to maximize the probability, or likelihood of  
 140 the target sequences.<sup>141</sup>

Let  $S$  be a particular sequence. Its Boltzmann probability is<sup>142</sup>

$$p(S) = \frac{1}{Z} \exp(-\beta \Delta G_S) \quad (3) \quad 143$$

where  $\Delta G_S = G_S^f - E_S^u$  is the folding free energy of  $S$ ,  $G_S^f$  is the  
 144 free energy of the folded form,  $\beta = 1/kT$  is the inverse  
 145 temperature, and  $Z$  is a normalizing constant (the partition  
 146 function). We then have<sup>147</sup>

$$\begin{aligned} kT \ln p(S) &= \sum_{i \in S} E^r(t_i) - G_S^f - kT \ln Z \\ &= \sum_{t \in \text{aa}} n_S(t) E_t^r - G_S^f - kT \ln Z \end{aligned} \quad (4) \quad 148$$

where the sum on the right is over the amino acid types and<sup>149</sup>  
 $n_S(t)$  is the number of amino acids of type  $t$  within the sequence<sup>150</sup>  
 $S$ .<sup>151</sup>

We now consider a set  $\mathcal{S}$  of  $N$  target sequences  $S$ ; we denote<sup>152</sup>  
 $\mathcal{L}$  the probability of the entire set, which depends on the model  
 parameters  $E_t^r$ ; we refer to  $\mathcal{L}$  as their likelihood.<sup>40</sup> We have<sup>153</sup>

$$\begin{aligned} kT \ln \mathcal{L} &= \sum_S \sum_{t \in \text{aa}} n_S(t) E_t^r - \sum_S G_S^f - NkT \ln Z \\ &= \sum_{t \in \text{aa}} N(t) E_t^r - \sum_S G_S^f - NkT \ln Z \end{aligned} \quad (5) \quad 154$$

where  $N(t)$  is the number of amino acids of type  $t$  in the whole  
 155 data set  $\mathcal{S}$ . The normalization factor or partition function  $Z$  is a  
 156 sum over all possible sequences  $R$ :<sup>157</sup>

$$\begin{aligned} Z &= \sum_R \exp(-\beta \Delta G_R) \\ &= \sum_R \exp(-\beta G_R^f) \prod_{t \in \text{aa}} \exp(\beta n_R(t) E_t^r) \end{aligned} \quad (6) \quad 158$$

In view of maximizing  $\mathcal{L}$ , we consider the derivative of  $Z$  with  
 159 respect to one of the  $E_t^r$ :<sup>160</sup>

$$\frac{\partial Z}{\partial E_t^r} = \sum_R \beta n_R(t) \exp(-\beta G_R^f) \prod_{s \in \text{aa}} \exp(\beta n_R(s) E_s^r) \quad (7) \quad 161$$

162 We then have

$$\frac{kT}{Z} \frac{\partial Z}{\partial E_t^r} = \frac{\sum_R n_R(t) \exp(-\beta \Delta G_R)}{\sum_R \exp(-\beta \Delta G_R)} = \langle n(t) \rangle \quad (8)$$

164 The quantity on the right is the Boltzmann average of the  
165 number  $n(t)$  of amino acids  $t$  over all possible sequences. In  
166 practice, this is the average population of  $t$  we would obtain in a  
167 long MC simulation. As usual in statistical mechanics,<sup>41</sup> the  
168 derivative of  $\ln Z$  with respect to one quantity ( $E_t^r$ ) is equal to  
169 the ensemble average of the conjugate quantity ( $\beta n_S(t)$ ).  
170 A necessary condition to maximize  $\ln \mathcal{L}$  is that its derivatives  
171 with respect to the  $E_t^r$  should all be zero. We see that

$$\frac{1}{N} \frac{\partial}{\partial E_t^r} \ln \mathcal{L} = \frac{1}{N} \sum_s n_s(t) - \langle n(t) \rangle = \frac{N(t)}{N} - \langle n(t) \rangle \quad (9)$$

172

173 so that

$$\mathcal{L} \text{ maximum} \Rightarrow \frac{N(t)}{N} = \langle n(t) \rangle, \quad \forall t \in \text{aa} \quad (10)$$

175 Thus, to maximize  $\mathcal{L}$ , we should choose  $\{E_t^r\}$  such that a long  
176 simulation gives the same amino acid frequencies as the target  
177 database.

178 **2.2. Searching for the Maximum Likelihood.** We will  
179 use two methods to approach the maximum likelihood  $\{E_t^r\}$   
180 values, starting from a current guess  $\{E_t^r(n)\}$ . With the first  
181 method, we step along the gradient of  $\ln \mathcal{L}$ , using the update  
182 rule:<sup>40</sup>

$$\begin{aligned} E_t^r(n+1) &= E_t^r(n) + \alpha \frac{\partial}{\partial E_t^r} \ln \mathcal{L} \\ &= E_t^r(n) + \delta E(n_t^{\text{exp}} - \langle n(t) \rangle_n) \end{aligned} \quad (11)$$

184 Here,  $n$  is an iteration number;  $\alpha$  is a constant;  $n_t^{\text{exp}} = N(t)/N$  is  
185 the mean population of amino acid type  $t$  in the target database;  
186  $\langle \cdot \rangle_n$  indicates an average over a simulation done using the  
187 current reference energies  $\{E_i^r(n)\}$ , and  $\delta E$  is an empirical  
188 constant with the dimension of an energy, referred to as the  
189 update amplitude. This update procedure is repeated until  
190 convergence. We refer to this method as the linear update  
191 method.

192 The second method, used previously,<sup>26,27</sup> employs a  
193 logarithmic update rule:

$$E_t^r(n+1) = E_t^r(n) - kT \ln \frac{\langle n(t) \rangle_n}{n_t^{\text{exp}}} \quad (12)$$

195 where  $kT$  is a thermal energy, set empirically to 0.5 kcal/mol (1  
196 cal = 4.184 J). We refer to this as the logarithmic update  
197 method. Both the linear and logarithmic update methods  
198 converge to the same optimum, specified by eq 10.

199 In the later iterations, some  $E_t^r$  values tended to converge  
200 slowly, with an oscillatory behavior. Therefore, we sometimes  
201 used a modified update rule, where the  $E_t^r(n+1) - E_t^r(n)$   
202 value computed with the linear or logarithmic method for  
203 iteration  $n$  was mixed with the value computed at the previous  
204 iteration, with the  $(n-1)$  value having a weight of  $1/3$  and the  
205 current value a weight of  $2/3$ . At each iteration, we typically ran  
206 500 million steps (per replica) of Replica Exchange Monte  
207 Carlo.

### 3. COMPUTATIONAL METHODS

208 **3.1. Effective Energy Function for the Folded State.** The energy matrix was computed with the following effective  
209 energy function for the folded state:  
210

$$\begin{aligned} E &= E_{\text{bonds}} + E_{\text{angles}} + E_{\text{dihedral}} + E_{\text{impr}} + E_{\text{vdw}} + E_{\text{Coul}} \\ &\quad + E_{\text{solv}} \end{aligned} \quad (13)$$

212 The first six terms in eq 13 represent the protein internal  
213 energy. They were taken from the Amber ff99SB empirical  
214 energy function,<sup>42</sup> slightly modified for CPD. The original  
215 backbone charges were replaced by a unified set, obtained by  
216 averaging over all amino acid types and adjusting slightly to  
217 make the backbone portion of each amino acid neutral.<sup>43</sup> The  
218 last term on the right of eq 13,  $E_{\text{solv}}$ , represents the contribution  
219 of solvent. We used a “Generalized Born + Surface Area”, or  
220 GBSA implicit solvent model:<sup>44</sup>

$$\begin{aligned} E_{\text{solv}} &= E_{\text{GB}} + E_{\text{surf}} = \frac{1}{2} \left( \frac{1}{\epsilon_W} - \frac{1}{\epsilon_p} \right) \\ &\quad \sum_{ij} q_i q_j (r_{ij}^2 + b_i b_j \exp[-r_{ij}^2/4b_i b_j])^{-1/2} + \sum_i \sigma_i A_i \end{aligned} \quad (14)$$

222 Here,  $\epsilon_W$  and  $\epsilon_p$  are the solvent and protein dielectric  
223 constants;  $r_{ij}$  is the distance between atoms  $i,j$  and  $b_i$  is the  
224 “solvation radius” of atom  $i$ .<sup>44,45</sup>  $A_i$  is the exposed solvent  
225 accessible surface area of atom  $i$ ;  $\sigma_i$  is a parameter that reflects  
226 each atom’s preference to be exposed or hidden from solvent.  
227 The solute atoms were divided into four groups with specific  $\sigma_i$   
228 values. The values were -5 (nonpolar), -40 (aromatic), -80  
229 (polar), and -100 (ionic) cal/mol/Å<sup>2</sup>. Hydrogen atoms were  
230 assigned a surface coefficient of 0. Surface areas were computed  
231 by the Lee and Richards algorithm,<sup>46</sup> implemented in the  
232 XPLOR program,<sup>47</sup> using a 1.5 Å probe radius. The MC  
233 simulations used a protein dielectric of  $\epsilon_p = 4$  or 8 (see  
234 Results).

235 In the GB energy term, the atomic solvation radius  $b_i$   
236 approximates the distance from  $i$  to the protein surface and is  
237 a function of the coordinates of all the protein atoms. The  
238 particular  $b_i$  form corresponds to a GB variant we call GB/  
239 HCT, after its original authors,<sup>44</sup> with model parameters  
240 optimized for use with the Amber force field.<sup>45</sup> Since  $b_i$   
241 depends on the coordinates of all the solute atoms,<sup>44</sup> an  
242 additional approximation is needed to make the GB energy  
243 term pairwise additive and to define the energy matrix.<sup>27,48</sup> We  
244 use a “Native Environment Approximation”, or NEA, in which  
245 the solvation radius  $b_i$  of each particular group (backbone, side  
246 chain or ligand) is computed ahead of time, with the rest of the  
247 system having its native sequence and conformation.<sup>27,48</sup>

248 The surface energy contribution  $E_{\text{surf}}$  is not pairwise additive  
249 either, because in a protein structure, surface area buried by one  
250 side chain may also be buried by another. To make this energy  
251 pairwise, we used the method of Street et al.<sup>49</sup> In this method,  
252 the buried surface of a side chain is computed by summing over  
253 the neighboring side chain and backbone groups. For each  
254 neighboring group, the contact area with the side chain of  
255 interest is computed, independently of other surrounding  
256 groups. The contact areas are then summed. To avoid  
257 overcounting the buried surface area, a scaling factor is applied  
258 to the contact areas involving buried side chains. Previous  
259 studies showed that a scaling factor of 0.65 works well.<sup>45,48</sup>

**3.2. Reference Energies in the Unfolded State.** In the CPD model, the unfolded state energy depends on the sequence composition through a set of reference energies  $E_t^r$  (eq 1). Here, the reference energies were assigned based on amino acid types  $t$ , taking into account also the position of each amino acid in the folded structure, through its buried or solvent-exposed character. Thus, for a given type (Ala, say), there were two distinct  $E_t^r$  values: a buried and an exposed value. This is so even though the reference energies are used to represent the unfolded, not the folded state. This procedure is supported by three assumptions. First, we assume residual structure is present in the unfolded state, so that amino acids partly retain their buried/exposed character. Second, we hypothesize that the unfolded state model compensates in a systematic way for errors in the folded state energy function, so that the folded structure contributes indirectly to the reference energies. Third, this strategy makes the model less sensitive to variations in the length of surface loops, and to the proportion of surface vs buried residues, which can vary widely among homologues (see below). As a result, the model should be more transferable within a protein family.

Distinguishing buried/exposed positions doubles the number of adjustable  $E_t^r$  parameters. Conversely, to reduce the number of adjustable parameters, we group amino acids into homologous classes (given in Results). Within each class  $c$ , and for each type of position (buried or exposed), the reference energies have the form

$$E_t^r = E_c^r + \delta E_t^r \quad (15)$$

Here,  $E_c^r$  is an adjustable parameter while  $\delta E_t^r$  is a constant, computed as the molecular mechanics energy difference between amino acid types within the class  $c$ , assuming an unfolded conformation where each amino acid interacts only with itself and with solvent. Specifically, we ran MC simulations of an extended peptide (the Syndecan1 peptide; see below) and computed the average energies for each amino acid type at each peptide position (excluding the termini). We took the differences between amino acid types and averaged them over the peptide positions. During likelihood maximization,  $E_c^r$  was optimized while  $\delta E_t^r$  was held fixed. To optimize the  $E_c^r$  values, we applied the linear or logarithmic method while the target frequencies corresponded to the experimental frequencies of the amino acid classes,  $n_c^{\text{exp}}$ , rather than of the individual types ( $n_t^{\text{exp}}$ , above).

**3.3. Experimental Sequences and Structural Models.** We considered the Tiam1 and Cask PDZ domains, whose crystal structures are known (PDB codes 4GVD and 1KWA,<sup>3</sup> respectively). They both belong to the class II binding motif, which recognizes the pattern  $\Phi\text{-X-}\Phi$  at the C-terminus of its peptide ligand, where  $\Phi$  is a hydrophobic amino acid. To define the target amino acid frequencies for likelihood maximization, we collected homologous sequences for each PDZ domain. We identified homologous sequences by using the Blast tool to search the Uniprot database, with the sequences taken from the PDB file as the query and the Blosum62 scoring matrix. We retained homologues with a sequence identity, relative to the query, above a 60% threshold and below an 85% threshold. If two homologues had a mutual sequence identity above 95%, one of the two was viewed as redundant and was discarded. This led to 50 Tiam1 and 126 Cask homologue sequences. The two sets of homologues are referred to as  $\mathcal{H}_T$  and  $\mathcal{H}_C$ , respectively. For each of the sets, say  $\mathcal{H}$ , we average over all homologues and all positions to obtain a computation of the

overall amino acid frequencies. The averaging is done separately for buried and exposed positions. The resulting amino acid frequencies are denoted  $\{f_t^b(\mathcal{H}), f_t^e(\mathcal{H})\}$ , where the subscript  $t$  represents an amino acid type and the superscripts b, and e refer to buried and exposed positions, respectively. Finally, the sets of mean frequencies derived from  $\mathcal{H}_T$  and  $\mathcal{H}_C$  were themselves averaged, giving the overall target amino acid frequencies,  $f_t^b = (f_t^b(\mathcal{H}_T) + f_t^b(\mathcal{H}_C))/2$  for each type  $t$ , and similarly for the exposed positions. Distinct target frequencies were thus obtained for buried and exposed positions.

Model parametrization and testing were mostly done for the apo state of each protein. However, for Cask, no apo X-ray structure was available at the beginning of this work, so a holo-like structure was used, where the peptide binding site is occupied by the C-terminus of another PDZ domain in the crystal lattice; the apo state was then modeled by removing this peptide. For the PDZ domain of Tiam1, we also used a holo structure then modeled the apo state by removing the peptide. For this PDZ domain, the backbone rms deviation between the apo and holo X-ray structures is just 0.5 Å; therefore, we expect the CPD model to be transferable between apo/holo Tiam1 states. For additional testing, we also considered two class I PDZ domains, syntenin and DLG2 (second PDZ domain in both cases), which recognize the pattern S/T-X-Φ at the C-terminus of its peptide ligand. Their X-ray structures are 1R6J and 2BYG, respectively. In both these structures, the peptide ligand was not cocrystallized, but the peptide binding site of each PDZ domain was partly occupied by the C-terminus of another protein molecule in the crystal lattice. The structures employed are listed in Table 1.

351 t1

**Table 1. Test Proteins**

protein name <sup>a</sup>	PDB code	residue numbers	no. active positions <sup>c</sup>
syntenin(2)	1R6J	192–273	72
DLG2(2)	2BYG	186–282	82
Cask	1KWA <sup>b</sup>	487–568	74
Tiam1	4GVD <sup>b</sup>	837–930	84

<sup>a</sup>In parentheses: number of the PDZ domain within the protein.

<sup>b</sup>Holo or holo-like structures. <sup>c</sup>The number of non-Gly, non-Pro positions, which can mutate during the design simulations.

To carry out the Monte Carlo design calculations, the structures were prepared and energy matrices were computed using procedures described previously.<sup>15,50</sup> Two missing segments in the Tiam1 PDZ domain (residues 851–854 and 868–869) were built using the Modeler program.<sup>51</sup> The peptide ligand was removed from the PDB structure for most of the design calculations before computing the energy matrix. For each pair of amino acid side chains, the interaction energy was computed after 15 steps of energy minimization, with the backbone held fixed and only the interactions of the pair with each other and the backbone included.<sup>26</sup> This short minimization alleviates the discrete rotamer approximation. Side chain rotamers were described by a slightly expanded version of the library of Tuffery et al.,<sup>52</sup> which has a total of 254 rotamers (summed over all amino acid types). This expanded library includes additional hydrogen orientations for OH and SH groups.<sup>48</sup> This rotamer library was chosen for its simplicity and because it gave very good performance in side chain

370 placement tests, comparable to the specialized Scwrl4 program  
 371 (which uses a much larger library).<sup>53,54</sup>

372 **3.4. Monte Carlo Simulations.** Sequence design was  
 373 performed with Proteus, which runs long Monte Carlo (MC)  
 374 simulations where selected amino acid positions can mutate  
 375 freely. The choice of mutating positions is user-defined and  
 376 depends on the specific design challenge. Four different choices  
 377 occurred in the present work. First, to optimize the reference  
 378 energies, we did simulations where about half of the positions  
 379 could mutate at a time. Second, the optimized models were  
 380 tested in simulations where all positions except Gly and Pro  
 381 were free to mutate. Hydrophobic titration of two PDZ  
 382 domains also employed this choice. Third, to produce designed  
 383 sequences to test through molecular dynamics, we did MC  
 384 simulations where Gly, Pro, and 11 positions closely involved in  
 385 peptide binding were held fixed, while all other positions were  
 386 allowed to mutate. Fourth, in the second Tiam1 application,  
 387 only four positions in the protein could mutate. In all these  
 388 cases (with two exceptions), mutations occurred randomly,  
 389 subject only to the MMGBSA energy function that drives the  
 390 simulation. In only two cases, an additional, “experimental”  
 391 energy term was used to explicitly bias the simulation to stay  
 392 close to the natural, Pfam sequences.

393 The Monte Carlo simulations used one- and two-position  
 394 moves, where either rotamers, amino acid types, or both  
 395 changed. For two-position moves, the second position was  
 396 selected among those that had a significant interaction energy  
 397 with the first (i.e., there was at least one rotamer conformation  
 398 where their unsigned interaction energy was 10 kcal/mol or  
 399 more). In addition, sampling was enhanced by Replica  
 400 Exchange Monte Carlo (REMC), where several MC simu-  
 401 lations (“replicas” or “walkers”) were run in parallel, at different  
 402 temperatures. Periodic swaps were attempted between the  
 403 conformations of two walkers  $i, j$  (adjacent in temperature).  
 404 The swap was accepted with the probability

$$405 \text{acc}(\text{swap}_{ij}) = \text{Min}[1, e^{(\beta_i - \beta_j)(\Delta E_i - \Delta E_j)}] \quad (16)$$

406 where  $\beta_i, \beta_j$  are the inverse temperatures of the two walkers and  
 407  $\Delta E_i, \Delta E_j$  are the changes in their folding energies due to the  
 408 conformation change.<sup>55,56</sup> We used eight walkers, with thermal  
 409 energies  $kT_i$  that range from 0.125 to 3 kcal/mol, spaced in a  
 410 geometric progression:  $T_{i+1}/T_i = \text{constant}$ .<sup>55</sup> Simulations were  
 411 done with the proteus program (which is part of the Proteus  
 412 package).<sup>27</sup> REMC was implemented with an efficient, shared-  
 413 memory, OpenMP parallelization.<sup>33</sup>

414 One simulation of Tiam1 and one of Cask were done that  
 415 included an “experimental”, biasing energy term, which  
 416 penalized sequences that had a low similarity to a reference,  
 417 experimental set. The bias energy had the form

$$418 \delta E_{\text{bias}} = c \sum_i (S_i^{\text{rand}} - S(t_i)) \quad (17)$$

419 where the sum extends over the amino acid positions  $i$ ;  $t_i$  is the  
 420 side chain type at position  $i$ ;  $S(t_i)$  is the (dimensionless)  
 421 Blosum40 similarity score versus the corresponding position in  
 422 the Pfam RPSS sequence alignment;  $S_i^{\text{rand}}$  is the mean score  
 423 (versus the same Pfam column) for a random type (where all  
 424 types are equiprobable), and  $c = 0.5$  kcal/mol.

425 **3.5. Rosetta Sequence Generation.** Monte Carlo  
 426 simulations were also performed using the Rosetta program  
 427 and energy function.<sup>37</sup> The simulations were done using  
 428 version 2015.38.58158 of Rosetta (freely available online),

429 using the command `fixbb -s Tiam1.pdb -resfile Tiam1.res -nstruct 10000 -ex1 -ex2 -linme- m_ig 10` where the ex1 and ex2 options activate an enhanced rotamer search for buried side chains, the last option (`linmem_ig`) corresponds to on-the-fly energy calculation, and default parameters were used otherwise. Gly and Pro residues present in the wildtype protein were not allowed to mutate, and positions that do mutate could not change into Gly or Pro (as with the Proteus design simulations). Simulations were run for each PDZ domain until 10 000 unique low energy sequences were identified, corresponding to run times of about 5 min per sequence on a single core of a recent Intel processor, for a total of 10 h (per protein) using 80 cores. This was comparable to the cost of the Proteus calculations (energy matrix plus Monte Carlo simulations).

430 **3.6. Sequence Characterization.** Designed sequences were compared to the Pfam alignment for the PDZ family, using the Blosum40 scoring matrix and a gap penalty of -6. This matrix is appropriate for comparing rather distant homologues (CPD and Pfam sequences in this case). Each Pfam sequence was also compared to the Pfam alignment, which allowed comparison between the designed sequences and a typical pair of natural PDZ domains. For these Pfam/Pfam comparisons, if a test PDZ domain  $T$  was part of the Pfam alignment, the  $T/T$  self-comparison was left out, to be more consistent with the designed/Pfam comparisons. The Pfam alignment was the “RPSS” alignment, consisting of 12 255 sequences. Similarities were computed separately for the 14 core residues and 16 surface residues, defined by their near-complete burial or exposure (listed in [Results](#)) and for the entire protein.

431 Designed sequences were submitted to the Superfamily library of Hidden Markov Models,<sup>57,58</sup> which attempts to classify sequences according to the Structural Classification of Proteins, or SCOP.<sup>59</sup> 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. The hmmscan program was executed using an E-value threshold of  $10^{-10}$  and a total of 15 438 models to represent the SCOP database.

432 To compare the diversity in the designed sequences with the diversity in natural sequences, we used the standard, position-dependent sequence entropy,<sup>60</sup> computed as follows:

$$433 S_i = - \sum_{j=1}^6 f_j(i) \ln f_j(i) \quad (18)$$

434 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 employed six residue classes, corresponding to the following groups: {LVIMC}, {FYW}, {G}, {ASTP}, {EDNQ}, and {KRH}. This classification was obtained by a cluster analysis of the BLOSUM62 matrix,<sup>61</sup> and by analyzing residue–residue contact energies in proteins.<sup>62</sup> To obtain a sense for how many amino acid types appeared at a typical position, we report the residue entropy in its exponential form,  $\exp(S)$  (which ranges from 1 to 6), averaged over the protein chain.

435 **3.7. Protein:Peptide Binding Free Energies.** For the Tiam1 PDZ domain, we used design calculations in the presence and absence of a bound peptide to obtain estimates of

489 the binding free energy differences between protein variants. If  
 490 a given sequence  $S$  was sampled in both the apo and holo  
 491 states, we computed the mean energy  $\langle E_{\text{holo}}(S) \rangle$ ,  $\langle E_{\text{apo}}(S) \rangle$  in  
 492 each of the two states by averaging over the sampled  
 493 conformations. Then, we took the difference

$$\Delta\Delta E(S, S') = (\langle E_{\text{holo}}(S') \rangle - \langle E_{\text{apo}}(S') \rangle) \\ - (\langle E_{\text{holo}}(S) \rangle - \langle E_{\text{apo}}(S) \rangle) \quad (19)$$

494

495 as our estimate of the binding free energy difference between  
 496 the variants  $S$  and  $S'$ . We also computed binding free energy  
 497 differences between groups of homologous sequences, say  $S$   
 498 and  $S'$ , by pooling the homologous sequences sampled in  
 499 either the apo or holo state, then averaging over the  
 500 conformations sampled and taking the energy difference  
 $\Delta\Delta E(S, S')$ .

501 **3.8. Molecular Dynamics Simulations.** Wildtype and a  
 502 quadruple mutant Tiam1 and 10 sequences designed with  
 503 Proteus were subjected to MD simulations with explicit solvent  
 504 and no peptide ligand. The starting structures were taken from  
 505 the MC trajectory or the crystal structure (wildtype protein and  
 506 quadruple mutant: PDB codes 4GVD and 4NXQ) and slightly  
 507 minimized with harmonic restraints to maintain the backbone  
 508 geometry. The protein was immersed in a large box of  
 509 nonoverlapping waters. The solvated system was truncated to  
 510 the shape of a truncated octahedral box using the Charmm  
 511 graphical interface or GUI.<sup>63</sup> The minimum distance between  
 512 protein atoms and the box was 15 Å and the final models  
 513 included about 11 000 water molecules. A few sodium or  
 514 chloride ions were included to ensure overall electroneutrality.  
 515 The protonation states of histidines were assigned to be neutral,  
 516 based on visual inspection. MD was done at room temperature  
 517 and pressure, using a Nose-Hoover thermostat and baro-  
 518 stat.<sup>64,65</sup> Long-range electrostatic interactions were treated with  
 519 a Particle Mesh Ewald approach.<sup>66</sup> The Amber ff99SB force  
 520 field and the TIP3P model<sup>67</sup> were used for the protein and  
 521 water, respectively. Simulations were run for 100–1000 ns,  
 522 depending on the sequence, using the Charmm and NAMD  
 523 programs.<sup>68,69</sup>

## 4. RESULTS

524 **4.1. Experimental structures and sequences.** Three  
 525 dimensional (3D) structures of the four test PDZ domains are  
 526 shown in Figure 1A. Fourteen core residues (identified visually)  
 527 superimposed well between the structures, while loops and  
 528 chain termini displayed large deviations. The Tiam1  $\alpha_2$  helix is  
 529 rotated slightly outward compared to the other three  
 530 structures.<sup>70</sup> Figure 1B illustrates the similarity between pairs  
 531 of PDZ domains, as determined by the rms deviation between  
 532 structurally aligned  $C_\alpha$  atoms and the pairwise sequence  
 533 identities. The rms deviations are between 1.0 and 2.1 Å and  
 534 the sequence identities between 17 and 33%. The Tiam1/Cask  
 535 sequence identity is 33% and their structural deviation is 1.7 Å  
 536 based on 42 aligned  $C_\alpha$  atoms. The syntenin and DLG2  
 537 structures are more similar, with a structural deviation of 1.0 Å  
 538 based on 60 aligned  $C_\alpha$  atoms.

539 Sequence conservation within the four PDZ domains and a  
 540 subset of the Pfam seed alignment is shown in Figure 2. The 14  
 541 positions used to define the hydrophobic core are highly,  
 542 although not totally conserved within the Pfam seed alignment.  
 543 Arginine, Lys, and Gln appear at some of the positions, since in  
 544 small proteins such as PDZ domains, the long hydrophobic

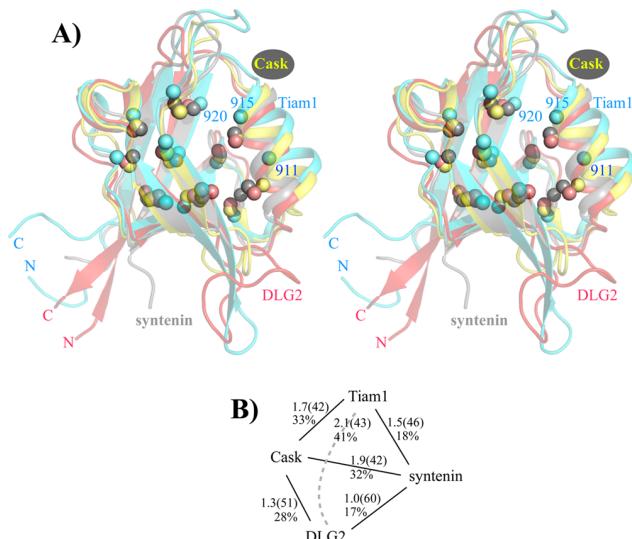
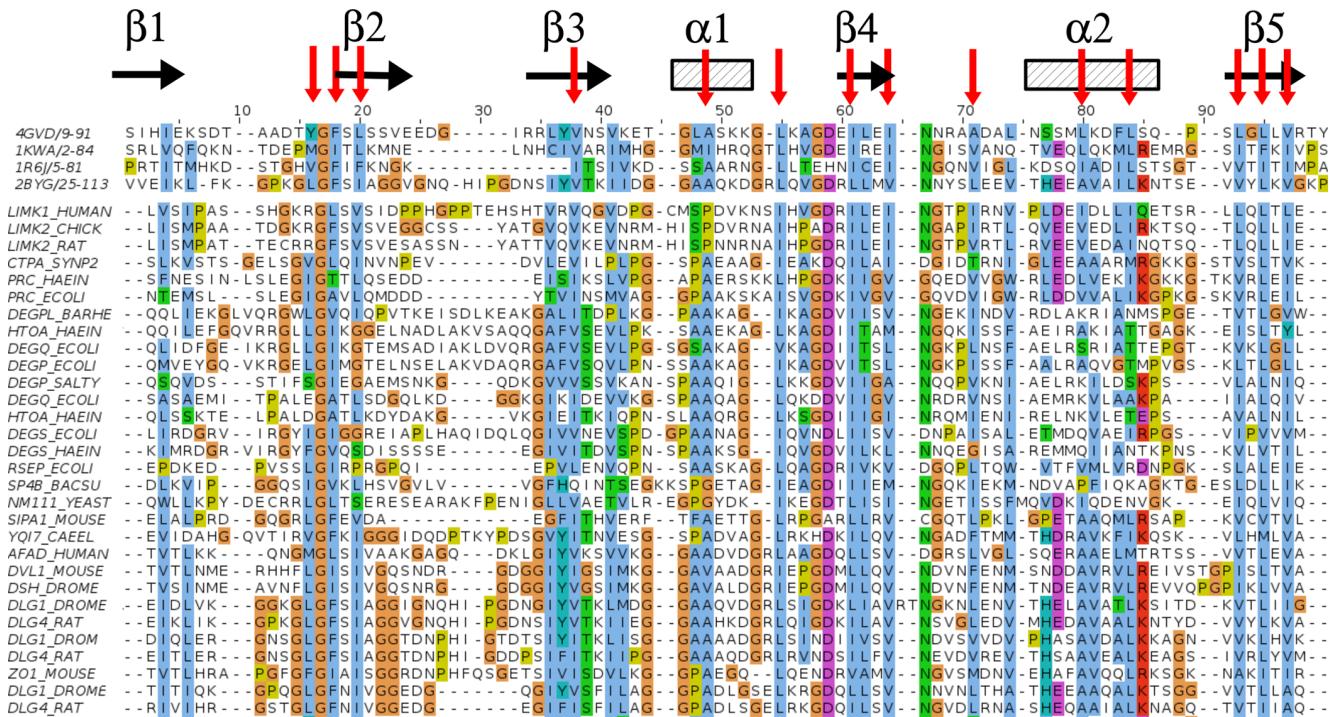


Figure 1. (A) Three dimensional view of four PDZ domains. The  $C_\beta$  atoms of 14 hydrophobic core residues are shown as spheres. Three core positions designed in this work are labeled (Tiam1 numbers). (B) Cluster representation of the PDZ domains studied. The links between domains are labeled with the percent identity scores and backbone rms deviations (Å); the number in parentheses is the number of aligned  $C_\alpha$  atoms used to compute the rms deviation.

portion of these side chains can be buried in the core while still allowing the polar tip of the side chain to be exposed to solvent. A few Asp and Glu residues also appear, in places where the sequence alignment may not reflect closely the 3D side chain superposition.

**4.2. Optimizing the Unfolded State Model.** We optimized the reference energies  $E_t^r$  for Tiam1 and Cask, using their natural homologues to define the target amino acid frequencies. The protein dielectric constant  $\epsilon_p$  was either 4 or 8. The  $E_t^r$  optimizations all converged to within 0.05 kcal/mol after about 20 iterations for most amino acid types, and to within 0.1 kcal/mol for the others (the weakly populated types), using either the linear or the logarithmic method (eq 11 or 12). Table 2 indicates the final reference energies. The  $E_t^r$  values were compared to, and agreed qualitatively with the energies computed from an extended peptide structure, which provides a less empirical model of the unfolded state. Table 3 compares the amino acid frequencies from the natural homologues and the simulations using parameters optimized with  $\epsilon_p = 8$ . Results obtained using parameters optimized with  $\epsilon_p = 4$  are given in Supporting Information. The theoretical population of the different amino acid classes agreed well with experiment, with rms deviations of about 1%, for both the exposed and buried positions. The agreement for the amino acid types was less good, with rms deviations of 3.9%/2.4% (buried/exposed positions). The intraclass frequency distributions depend explicitly on the energy offsets  $\delta E_t^r$  defined within each class, which were computed with molecular mechanics (see Methods, eq 15).

**4.3. Assessing Designed Sequence Quality. Family Recognition Tests.** Proteus design simulations used Replica Exchange Monte Carlo with eight replicas and 750 million steps per replica, at thermal energies  $kT$  that ranged from 0.125 to 3 kcal/mol. All positions (except Gly and Pro) were allowed to mutate freely into all amino acid types except Gly and Pro. The simulations were done with the MMGBSA energy function,



**Figure 2.** Alignment of natural PDZ sequences. The top four sequences were tested in this work. The others are the first 30 sequences from the Pfam seed alignment. Fourteen hydrophobic core positions are indicated by red arrows and the secondary structure elements are shown for reference. The Clustal color scheme is used, in which conserved amino acids are colored according to their physical chemical properties.

**Table 2. Unfolded State Reference Energies  $E_t^r$  (kcal/mol)**

residues	peptide <sup>a</sup>	design, $\epsilon_p = 8$		design, $\epsilon_p = 4$
		buried	exposed	
ALA	0.00	0.00	0.00	0.00
CYS	-0.85	-0.85	-0.85	-0.60
THR	-5.44	-5.44	-5.44	-8.22
SER	-6.43	-3.71	-4.74	-5.68
ASP	-17.28	-11.90	-15.88	-20.31
GLU	-17.35	-11.97	-15.95	-17.67
ASN	-12.25	-7.82	-10.22	-17.70
GLN	-11.50	-7.07	-9.47	-14.35
HIS <sup>b</sup>	9.02	12.53	9.73	13.52
HIS <sub>e</sub> <sup>b</sup>	6.98	10.49	7.69	9.90
HIS <sub>d</sub> <sup>b</sup>	7.35	10.86	8.06	10.62
ARG	-36.90	-32.00	-35.18	-54.08
LYS	-11.71	-6.76	-10.17	-8.41
ILE	4.22	4.63	3.63	5.30
VAL	-0.15	0.26	-0.74	-0.89
LEU	-0.53	-0.12	-1.12	-0.97
MET	-1.78	-2.05	-2.40	-1.11
PHE	-3.98	-0.23	-4.17	0.66
TRP	-5.96	-2.21	-6.15	0.17
TYR	-10.09	-5.80	-9.82	-7.87

<sup>a</sup>Energies within an extended peptide structure (averaged over positions). <sup>b</sup>His protonation states.

assigned to the correct family: 91% for Tiam1 and 100% for s88 Cask, with E-values of around  $10^{-3}$  for the family assignments. s89 These values are similar to Rosetta (90 and 98% family s90 recognition for Tiam1 and Cask). Changing the protein s91 dielectric constant to  $\epsilon_p = 4$  gave somewhat poorer results s92 for Tiam1, with 53% of the sequences designed with Proteus s93 correctly recognized by Superfamily. s94

**Sequences and Sequence Diversity.** Tiam1 and Cask s95 sequences predicted by Proteus and Rosetta as well as natural s96 sequences are shown in Figure 3 for the 14 core residues and in s97 f3 Figure 4 for the 16 surface residues (Tiam1 only). The s98 f4 sequences are represented as sequence logos. As seen in many s99 previous CPD studies,<sup>30,72</sup> agreement with experiment for the s100 core residues is very good, while agreement for the surface s101 residues is much poorer. The behavior of surface positions was s102 also probed by designing each position individually, with the s103 rest of the protein free to explore rotamers but not mutations s104 (“mono-position” design). The corresponding logo (Figure 4) s105 shows an excess of Arg and Lys residues, suggesting that the s106 CPD reference energies are not yet fully optimal, despite the s107 extensive empirical  $E_t^r$  tuning. Sequence similarity scores are s108 given in the next subsection. s109

The diversity of the natural and designed sequences was s110 characterized by a mean, exponential sequence entropy (see s111 Methods), which corresponds to a mean number of sampled s112 sequence classes per position. For example, a value of 2 at a s113 particular position indicates that amino acids from two of the s114 six classes are present at that position within the set of analyzed s115 sequences. An overall average value of two indicates that on s116 average, two amino acid classes are present at any position s117 within the analyzed sequences. For reference, the Pfam RPSS s118 set of 12 255 natural sequences has a mean entropy of 3.4. s119 Pooling the designed Tiam1 and Cask sequences gave an s120

without any bias toward natural sequences or any limit on the s82 number of mutations. The 10 000 sequences with the lowest s83 energies among those sampled by any of the MC replicas were s84 retained for analysis, along with the 10 000 Rosetta sequences. s85 These sequences were analyzed by the Superfamily fold s86 recognition tool<sup>58,71</sup> (Table 4). With a protein dielectric s87 constant of 8, we obtained a high percentage of sequences

**Table 3.** Amino Acid Composition (%) of Natural and Designed PDZ Proteins<sup>a</sup>

type	natural sequences				designed sequences			
	buried		exposed		buried		exposed	
type	type	class	type	class	type	class	type	class
A	5.9		4.6		4.1		7.2	
C	1.5	11.2	1.2	13.4	8.6	12.7 [1.5]	5.8	13.6 [0.2]
T	3.8		7.6		0.0		0.6	
S	4.7	4.7	10.2	10.2	4.9	4.9 [0.2]	10.7	10.7 [0.5]
D	3.5		6.2		7.4		8.0	
E	6.1	9.6	10.5	16.7	2.0	9.4 [-0.2]	8.1	16.1 [-0.6]
N	1.9	2.7	7.4		1.8		8.6	
Q	0.8		8.7	16.1	1.0	2.8 [0.1]	8.5	17.1 [1.0]
H <sup>+</sup>	0.7		4.7		0.1		1.8	
H <sub>e</sub>	0.0	0.7	0.0	4.7	0.6	0.9 [0.2]	2.2	4.5 [-0.2]
H <sub>d</sub>	0.0		0.0		0.2		0.5	
I	15.7		4.1		25.1		8.4	
V	13.5	49.6	5.5	14.4	12.8	46.7 [-2.9]	3.3	15.3 [0.9]
L	20.4		4.8		8.8		3.6	
M	5.0	5.0	1.4	1.4	5.9 [0.9]	5.9	1.4 [0.0]	1.4
K	6.5	6.5	10.1	10.1	5.5	5.5 [-1.0]	10.8	10.8 [0.7]
R	1.8	1.8	9.5	9.5	2.2	2.2 [0.4]	9.1	9.1 [-0.4]
F	5.0	5.0	0.4		3.2		0.3	
W	0.0		0.0	0.4	2.3	5.5 [0.5]	0.2	0.5 [0.1]
Y	2.9	2.9	0.9	0.9	3.4	3.4 [0.5]	0.9	0.9 [0.0]
G	0.0		1.7		0.0		0.0	
P	0.3	0.3	0.4	2.1	0.0	0.0 [-0.3]	0.0	0.0 [-2.1]

<sup>a</sup>Compositions are given for buried/exposed positions, for individual amino acid types (left) and for classes (right); values in brackets (right) are the deviations between design and experiment per class. The experimental target set included the Tiam1 and Cask homologues.

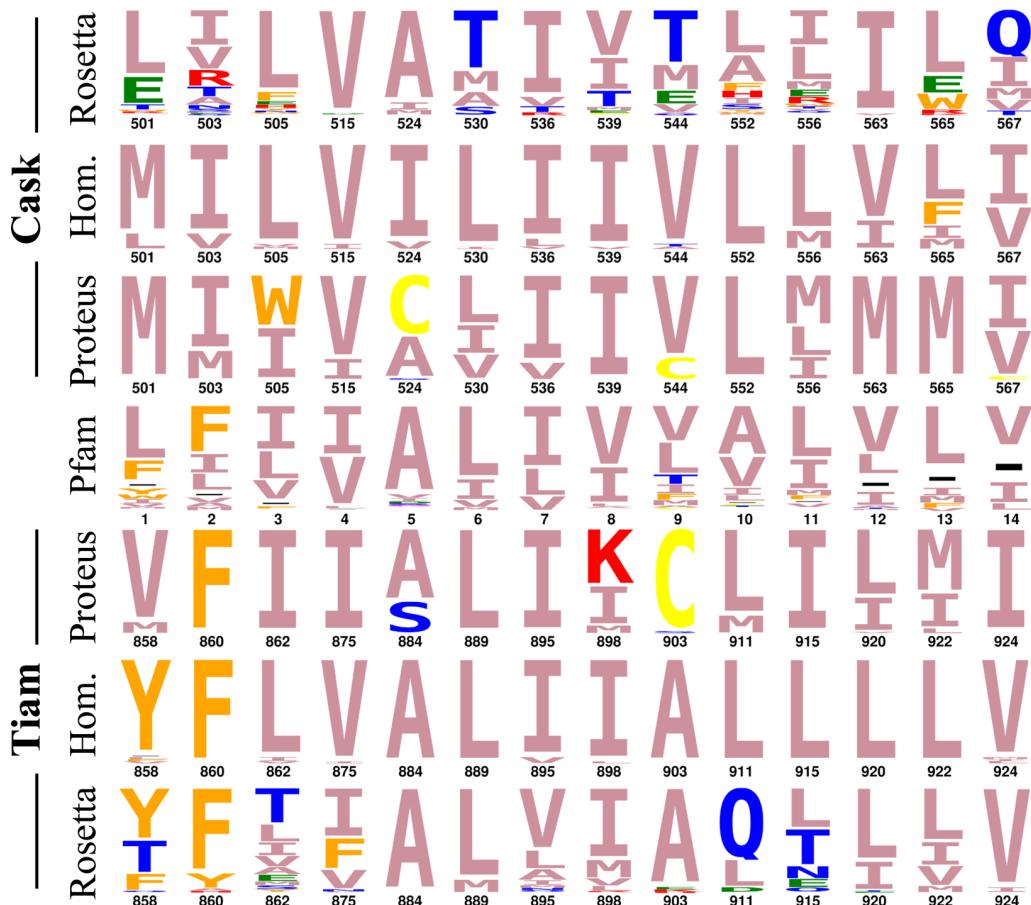
**Table 4.** Fold Recognition of Designed Sequences by Superfamily

protein	design model	match/seq length <sup>a</sup>	superfamily		family	
			E-value <sup>b</sup>	success no. <sup>c</sup>	E-value <sup>b</sup>	success no. <sup>c</sup>
Tiam1	Proteus, $\epsilon_p = 4$	53/94	$1.0 \times 10^{-4}$	10000	$7.0 \times 10^{-2}$	5259
Cask	Proteus, $\epsilon_p = 4$	76/83	$5.1 \times 10^{-7}$	10000	$1.6 \times 10^{-2}$	10000
syntenin	Proteus, $\epsilon_p = 8$	69/91	$1.3e \times 10^{-2}$	9999	$4. \times 10^{-3}$	9999
DLG2	Proteus, $\epsilon_p = 8$	85/97	$8.0 \times 10^{-9}$	10000	$5.0 \times 10^{-3}$	10000
Tiam1	Proteus, $\epsilon_p = 8$	64/94	$1.2 \times 10^{-4}$	9920	$5.2 \times 10^{-2}$	9058
Cask	Proteus, $\epsilon_p = 8$	71/83	$3.2 \times 10^{-7}$	10000	$8.2 \times 10^{-3}$	10000
Tiam1	Rosetta	65/94	$4.4 \times 10^{-4}$	9035	$2.8e \times 10^{-2}$	9030
Cask	Rosetta	68/83	$2.8 \times 10^{-5}$	9832	$7.5 \times 10^{-3}$	9832
syntenin	Rosetta	76/82	$7.3 \times 10^{-13}$	10000	$1.8 \times 10^{-3}$	10000
DLG2	Rosetta	86/97	$1.3 \times 10^{-9}$	10000	$9.6 \times 10^{-4}$	10000

<sup>a</sup>The average match length for sequences recognized by Superfamily and the total sequence length. <sup>b</sup>Average E-values for superfamily assignments to the correct SCOP superfamily/family. <sup>c</sup>The number of designed sequences (out of 10000 tested) assigned to the correct SCOP superfamily/family.

entropy of 2.2 with Rosetta and 2.0 with Proteus, indicating that these two backbone geometries cannot accommodate as much diversity as the much larger RPSS set. Taking the 10 000 lowest energy sequences sampled with the room temperature Monte Carlo replica (instead of the 10 000 lowest energies

sampled collectively by all replicas at all temperatures) and pooling Tiam1 and Cask as before gave a higher overall entropy of 2.9 with Proteus. With Rosetta, entropy in the core was only slightly below the average over all positions. With Proteus, it



**Figure 3.** Sequence logos for the conserved hydrophobic core of designed and natural Tiam1 and Cask sequences. “Hom.” corresponds to the homologues that make up our target set of sequences (used for  $E_t$  optimization). “Pfam” corresponds to the Pfam seed alignment. Proteus sequences were generated with model  $\epsilon_p = 8$ . The height of each letter is proportional to the abundance of each type at the corresponding position in the Proteus/Rosetta simulations or the natural sequences. The color of each letter is determined by the physical chemical properties of each amino acid type.

630 was distinctly lower (1.25). For the Pfam-RP55 sequences, it  
631 was 1.8.

632 **Blosum Similarity Scores.** Figure 5 shows the computed  
633 Blosum40 similarity scores between designed and natural  
634 sequences. With Proteus, for both Tiam1 and Cask, the overall  
635 similarities overlapped with the bottom of the peak of the  
636 natural scores, and were comparable to the values for the  
637 Rosetta sequences. For the surface residues, shown separately,  
638 similarity to the natural sequences was low (scores below zero),  
639 both for Proteus and Rosetta. With a protein dielectric constant  
640 of 4, Proteus performed about as well as with  $\epsilon_p = 8$ , giving  
641 almost the same similarity averaged over all Tiam1 and Cask  
642 positions, for example.

643 While the similarity scores vs Pfam with Proteus were  
644 comparable to Rosetta (Figure 5), the identity scores vs the  
645 wildtype sequence were significantly higher with Rosetta.  
646 Identity scores excluding (respectively, including) Gly and  
647 Pro positions (which did not mutate) were 20% (28%) for  
648 Proteus vs 26% (34%) for Rosetta. Evidently, for Tiam1 and  
649 Cask, Rosetta performed  $\approx 5$  fewer mutations than Proteus.

650 For certain applications, we may need to specifically explore a  
651 sequence space region very similar to Pfam, beyond the  
652 similarity provided by an MMGBSA energy. This can be  
653 achieved by adding to the energy an “experimental,” or bias  
654 energy term that explicitly favors high sequence scores. Figure 5

includes results that used such a biased energy term: by 655 construction, it led to very high similarity scores. A bias energy 656 term could also be used to limit the total number of mutations. 657

658 **4.4. Cross-Validation Tests.** As a first cross-validation test,  
659 we applied the reference energies optimized using Tiam1 and  
660 Cask homologues (with  $\epsilon_p = 8$ ) to two other PDZ domains:  
661 DLG2 and syntenin. The superfamily scores were comparable  
662 to those obtained for Tiam1 and Cask, with 100% family  
663 recognition (Table 4). Sequences designed with Rosetta for  
664 DLG2 and syntenin also gave 100% family recognition. For  
665 further cross-validation, we optimized reference energies using  
666 an alternate set of PDZ domains: DLG2, syntenin, PSD95,  
667 GRIP, INAD, and NHERF. Target frequencies were defined by  
668 a small set of their natural homologues. We used  $\epsilon_p = 8$ . To  
669 distinguish the new and initial model variants, we refer to the  
670 new variant as the  $n = 6$  model (it uses six PDZ domains for  
671 parametrization), and the initial model as the “T+C” model (it  
672 used Tiam1 and Cask). The new,  $n = 6$  reference energies were  
673 then used to produce designed Tiam1 and Cask sequences,  
674 which were subjected to Superfamily tests and similarity  
675 calculations. The Superfamily performance for Tiam1 was  
676 slightly degraded, compared to the previous, T + C model. The  
677 Tiam1 Superfamily score decreased from 90.6% to 76.6% for  
678 family recognition. The Cask score was unchanged. Histograms  
679 of Blosum similarity scores (Supporting Information) show that  
680



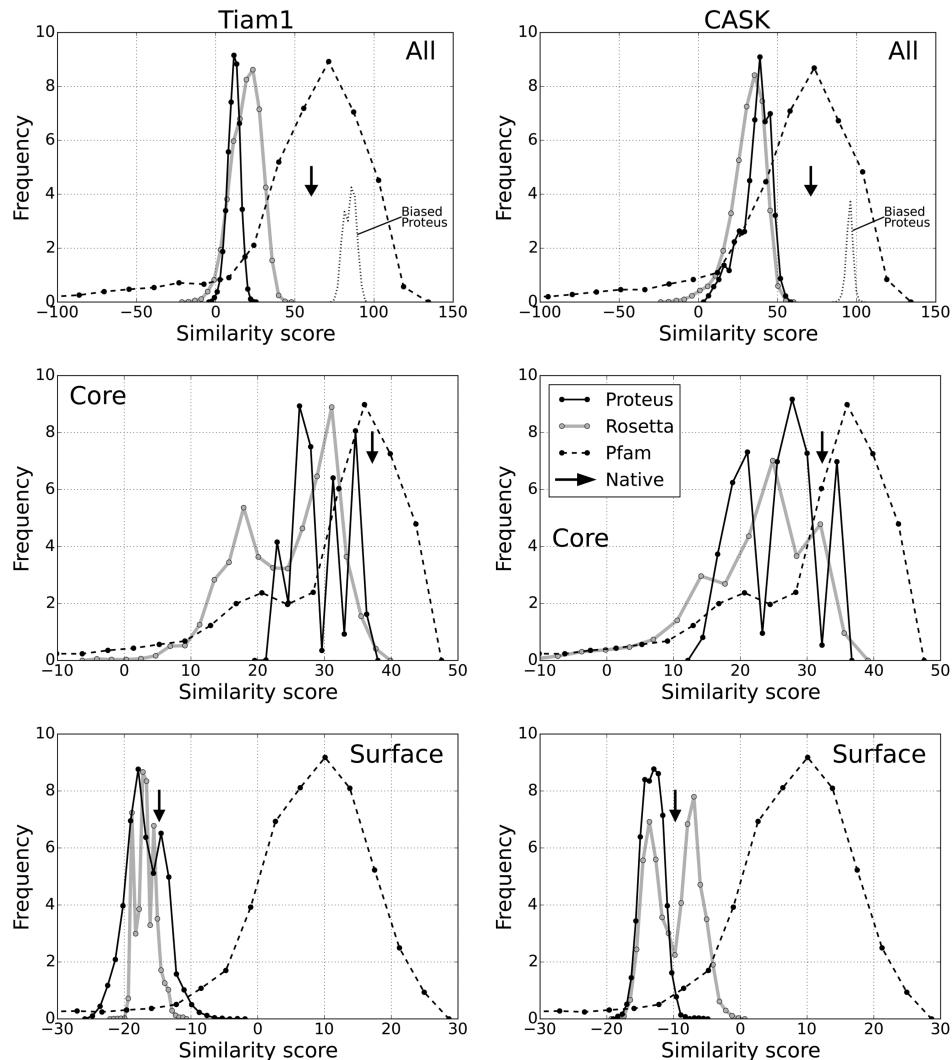
**Figure 4.** Sequence logos for 16 surface positions in Tiam1. Same representation as Figure 3. The “mono-position” results are from a set of simulations where only one amino acid at a time could mutate, the rest of the protein having its native sequence (see text). Amino acid colors as in Figure 3.

the overall scores for Tiam1 and Cask with  $n = 6$  were very similar to the T + C model, while the scores for the core positions were actually shifted to higher, not lower values. For DLG2 and syntenin, we also computed similarity scores using both the initial, T+C parametrization and the new,  $n = 6$  parametrization. The similarity scores with the T+C model were slightly poorer than with the  $n = 6$  model, as expected. The overall score decreased by about 20 points for syntenin and about 10 points for DLG2 (Supporting Information). Overall, the cross-validated models degraded performance slightly. Thus, for any PDZ domain of interest, it may be preferable to optimize reference energies specifically for that domain, rather than transferring values parametrized using other PDZ domains.

**4.5. Stability of Designed Sequences in Molecular Dynamics Simulations.** As another test of the design model, 10 Tiam1 sequences designed with Proteus were subjected to molecular dynamics simulations (MD) using an explicit solvent environment. These sequences were obtained using Proteus with either  $\epsilon_p = 8$  or the less polarizable value  $\epsilon_p = 4$ . Although no peptide ligand was present during the design simulations, 11 positions in the binding pocket that make close contact with the peptide when it is present were not allowed to mutate. This was done to allow future experimental testing of designed sequences by a peptide binding assay. Among the 2500 lowest energy designed sequences, we narrowed down the choice of sequences using the following four criteria: (a) sequences should have a nonneutral isoelectric point, (b) they should be assigned to the correct SCOP family by Superfamily with good  $E$ -values, (c) they should have good Pfam similarity scores, and (d) they should have at most 15 mutations that drastically change the amino acid type compared to the wildtype protein (such a change is defined by a Blosum62 similarity score between the two amino acid types of  $-2$  or less). Applying these criteria reduced the number of sequences to 66 from the

$\epsilon_p = 8$  model and 45 from the  $\epsilon_p = 4$  model. In addition, we eliminated sequences that had two mutations that created a buried cavity and those that had net protein charges of  $+6$  or more (which could lead to protein instability). A total of six sequences were chosen for further analysis. We refer to them as sequences 1–6 or seq-1, ..., seq-6. Sequences 1, 2, 4, and 5 were modified further manually to eliminate charged residues in the exposed loop 852–856 (lysines were changed manually to alanine), giving sequences 1', 2', 4', and 5'. The 10 sequences are shown in Figure 6A. Using these sequences as queries to search Uniprot with Blast, the top hits were either Tiam1 mammalian orthologs (including human Tiam1) or uncharacterized proteins, with identity scores between 35 and 40% and Blast  $E$ -values of around  $10^{-8}$ – $10^{-7}$  (except for one sequence which gave hits with lower  $E$ -values of around  $10^{-10}$ ).

All 10 sequences were subjected to MD simulations with explicit solvent. Initial simulations were run for 100 ns, with all 10 sequences exhibiting good stability. Six were extended to lengths of 500 or 1000 ns. The wildtype protein (WT) was also simulated for 1000 ns. The WT sequence appeared stable over the entire simulation, judging by its rms deviations from the WT X-ray structure and from its own mean MD structure (Figure 6B). The mean MD structure had a backbone rms deviation of 1.0 Å from the WT X-ray structure (excluding 3–4 residues at each terminus and one very flexible loop, residues 850–857). During the MD trajectory, the rms deviation from the mean MD structure varied in the range 1–1.5 Å, without any visible drift (Figure 6B). A weakly stable quadruple mutant (QM) with an unfolding free energy of just 1 kcal/mol<sup>70</sup> was also simulated for 1000 ns. The mean MD structure of QM had a backbone rms deviation of 1.6 Å relative to the QM X-ray structure (4NXQ). Note that the X-ray structure includes a peptide ligand, whereas the MD simulation represents the apo state. The average MD structure of QM (Figure 6C) exhibited some unwinding of the N-terminus of the  $\alpha_2$  helix. During the



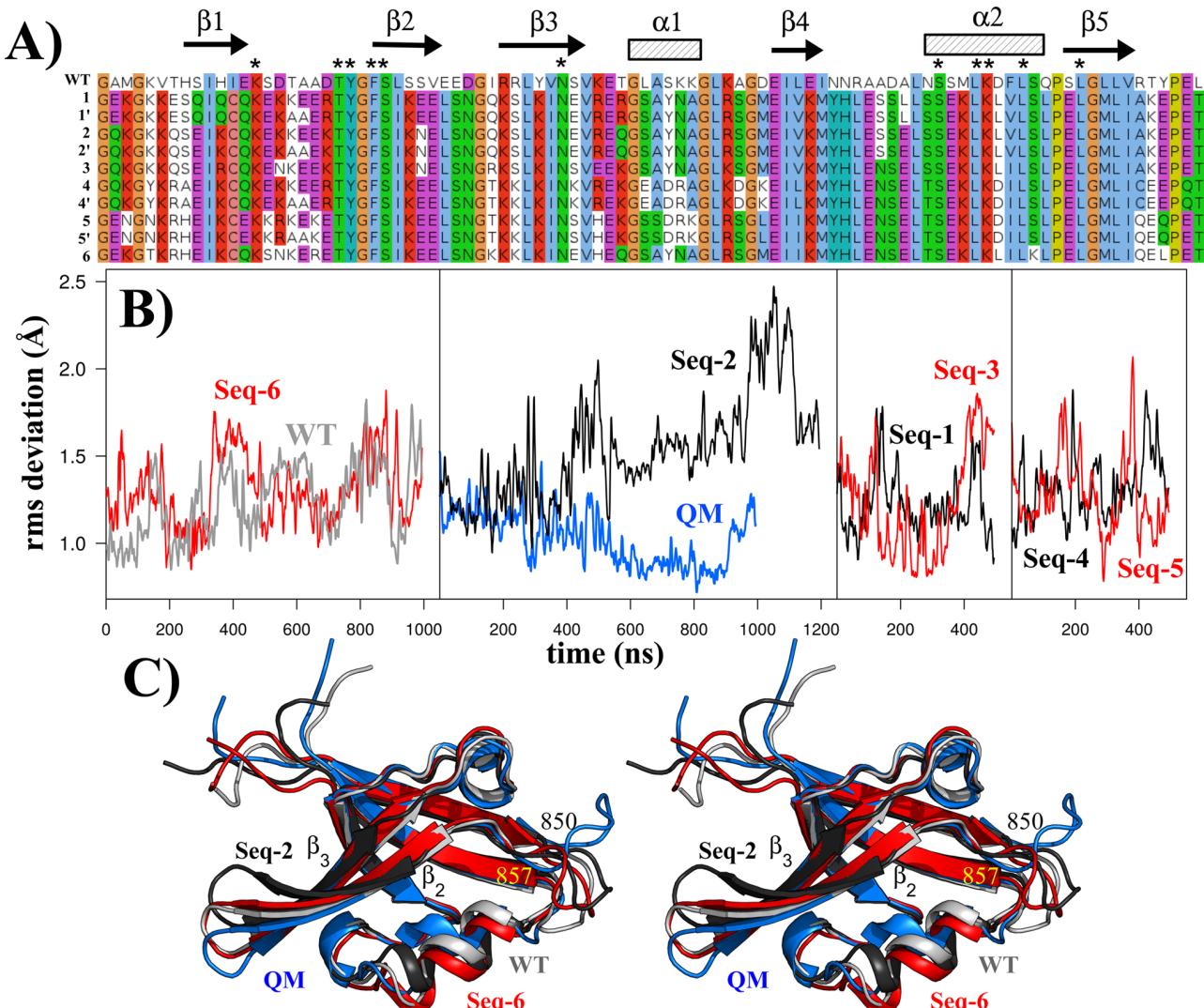
**Figure 5.** Histogram plots showing similarity scores for designed PDZ sequences. Similarity scores for Tiam1 (left) and Cask (right), relative to the Pfam-RPSS alignment. The scores were computed for all positions (top), 14 core positions (middle), and 16 surface positions (bottom). Values are shown for Proteus, Rosetta, and Pfam sequences (all compared to RPSS). The similarity score of the wildtype sequence is indicated in each panel by a vertical arrow. The top panels include results for Proteus simulations where a bias energy was included, which explicitly favors sequences that are similar to Pfam (dotted lines, labeled “Biased Proteus”). Notice that the designed sequences represented in each top, middle, or bottom panel were the same; only the positions included in the similarity score calculation differ between panels.

QM simulation, the structure had deviations from its average MD structure in the 0.8–1.2 Å range (Figure 6B) and appeared stable.

Sequences 1, 3, 4, and 5 were simulated for 500 ns; sequence 3 moved away from the mean MD structure toward the end of the simulation; the other three sequences appeared stable (Figure 6B). Sequence 2 (or seq-2) appeared stable up to almost 1000 ns (Figure 6B). The mean seq-2 structure exhibited a shortening of the  $\beta$  strands 2 and 3. Note that in the holo state, strand 2 makes direct contact with the peptide ligand. The rms deviations between the average MD structure of seq-2 and the WT and QM X-ray structures were 1.5 and 1.6 Å, respectively. During the seq-2 MD simulation, the rms deviation of seq-2 from its average MD structure varied in the range 1.3–2 Å up to almost 1000 ns. At this point, just before 1000 ns, seq-2 underwent a large fluctuation. When the simulation was extended for another 100 ns, the fluctuation largely regressed. More data are needed to determine if this fluctuation signals instability of this designed sequence.

Sequence 6 appeared stable throughout the microsecond MD simulation (Figure 6B). Its mean MD structure had a backbone rms deviation from the WT X-ray structure of just 1.0 Å, the same deviation as the mean WT MD structure. The mean MD structures of seq-6 and WT are superimposed in Figure 6C and are very similar to each other, with a 1.2 Å backbone deviation between them. During the seq-6 MD trajectory, the deviations of seq-6 away from its mean MD structure fluctuated between about 1 and 1.5 Å, without any visible drift over the microsecond MD simulation.

**4.6. Application: Growing the PDZ Hydrophobic Core.** As a first application of our optimized models, we examined the designability of the Tiam1 and Cask hydrophobic cores. Each PDZ domain was subjected to Replica Exchange Monte Carlo simulations with a succession of biased energy functions that increasingly favored hydrophobic residues. The first simulation included a bias energy term  $\delta = 0.4$  kcal/mol (per position) that penalized hydrophobic amino acid types (ILMVAWFY). The final simulation included a bias energy term  $\delta = -0.4$  kcal/mol that favored hydrophobic amino acids.



**Figure 6.** MD simulations of Tiam1 variants designed with Proteus. (A) Sequences of the wildtype (WT) PDZ domain and the 10 designed variants simulated by MD. Asterisks indicate peptide binding residues held fixed during the design simulations. (B) Backbone rms deviations over the course of an MD simulation for WT, QM, and six designed variants relative to the corresponding mean MD structure. (C) Mean MD structures of WT, QM, seq-6, seq-2, seq-1, seq-3, seq-4, and seq-5.

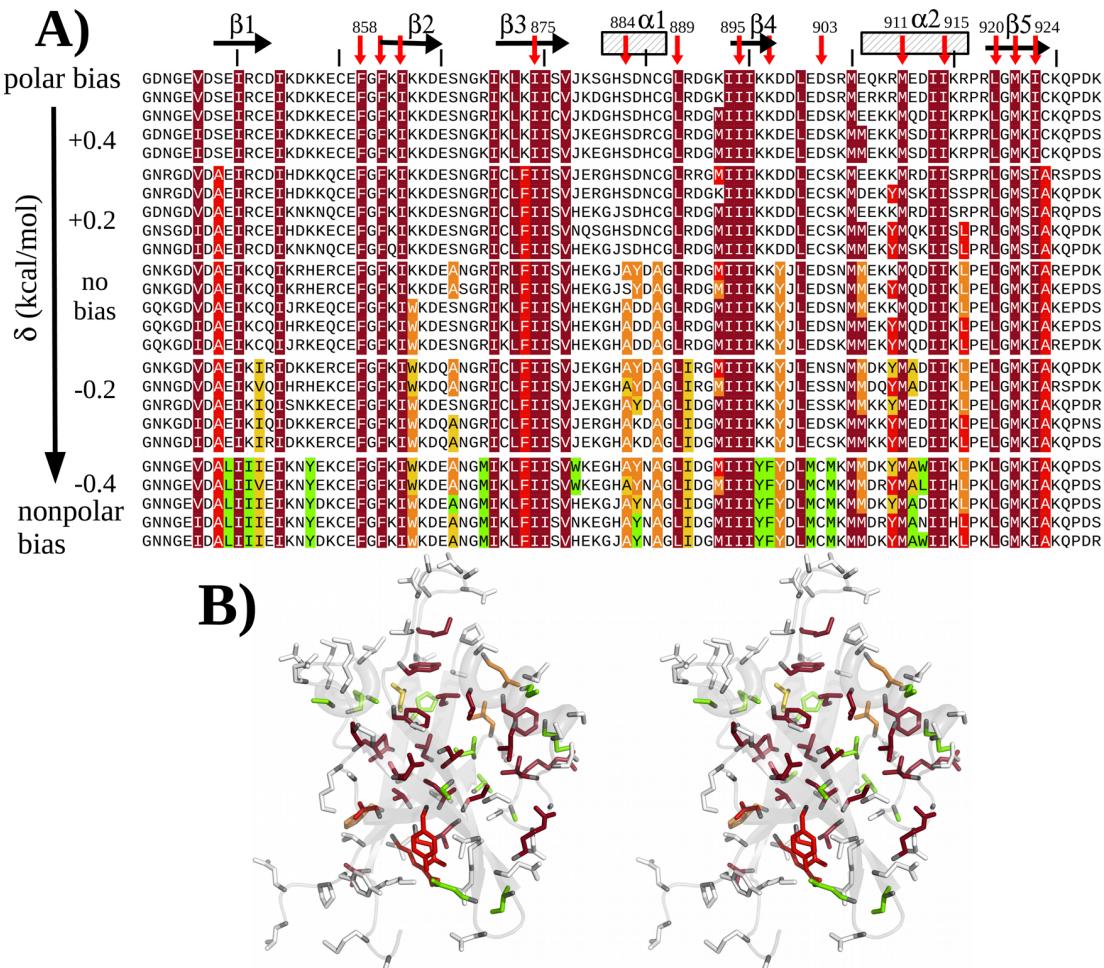
788 mol (per position) that favored hydrophobic types. Inter-  
789 mediate bias energy values  $\delta = 0.2, 0$ , and  $-0.2$  kcal/mol were  
790 also simulated. By gradually decreasing the bias energy value  $\delta$ ,  
791 we effectively “titrate in” hydrophobic residues.

792 The results for Tiam1 are illustrated in Figure 7. At the  
793 largest  $\delta$  value, the Tiam1 hydrophobic core was depleted, with  
794 10 amino acid positions (out of 94) changed to polar types.  
795 The changed positions mostly lie on the outer edge of the core.  
796 At the intermediate  $\delta$  values, the hydrophobic core remained  
797 native-like. At the most negative  $\delta$  value, the hydrophobic core  
798 became larger, expanding out toward surface regions, with 14  
799 polar positions changed to hydrophobic types. Thus, the  
800 numbers of positions changed were approximately symmetric  
801 (around  $\pm 12$  changes), reflecting the bias. About 2/3 of the  
802 changes were in secondary structure elements. Overall, the  
803 observed propensities of each position to become polar or  
804 hydrophobic in the presence of a large or small penalty bias  
805 energy  $\delta$  can be thought of as a hydrophobic designability  
806 index. Here, 11 of the 14 conserved core positions (all but  
807 positions 884, 898, and 903) remained hydrophobic even at the

808 highest level of polar bias, along with 13 other positions,  
809 indicating that these positions have the highest hydrophobic  
810 propensity. Furthermore, 14 positions changed from polar to  
811 hydrophobic at the highest bias level, indicating that these  
812 positions also have a certain hydrophobic propensity. Results  
813 for Cask were similar, with 11 positions changed to polar at the  
814 highest polar bias and 9 changed to hydrophobic at the highest  
815 hydrophobic bias.

We also derived a parameter to describe the relative number  
816 of amino acid type changes per unit bias energy. This parameter  
817 was defined as the number  $\delta N$  of residue positions changed  
818 from nonpolar to polar, divided by the product of the change  
819  $\delta E$  in bias energy and the mean number  $N$  of nonpolar  
820 positions at zero bias. We call it the hydrophobic susceptibility,  
821  $\chi_h$ . For the Tiam1 PDZ domain, this calculation amounts to  
822  $\chi_h = \frac{1}{N} \frac{\delta N}{\delta E} = 0.88$  changes (per position) per kcal/mol. For  
823 Cask, the susceptibility was  $\chi_h = 0.71$  changes per kcal/mol.

**4.7. Application to Tiam1: Designing Specificity**  
824 **Positions.** As a second application, we redesigned four  
825

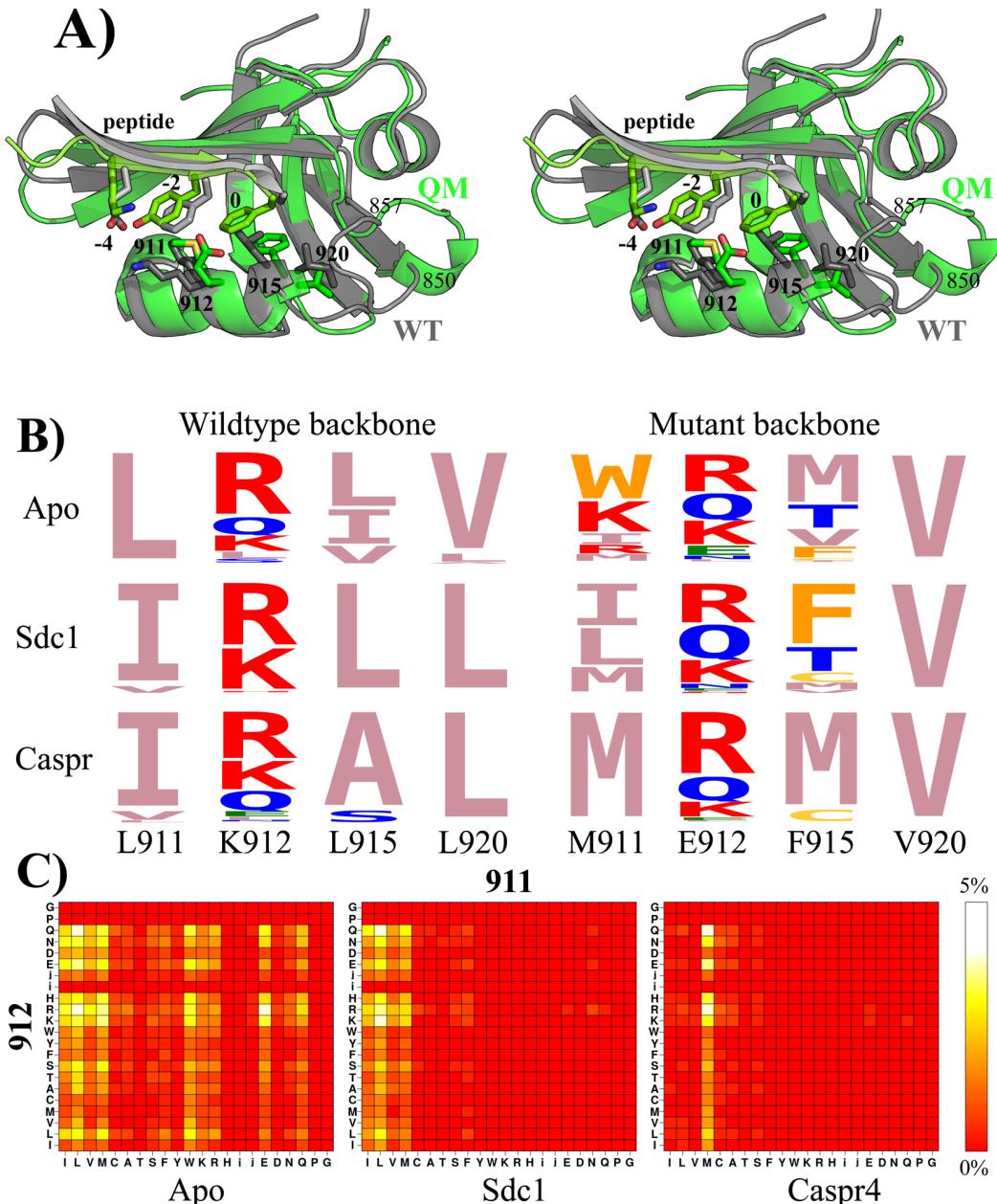


**Figure 7.** (A) Tiam1 sequences designed with different levels of hydrophobic bias. The top (respectively, bottom) sequences were obtained in the presence of a bias  $\delta$  opposing (favoring) hydrophobic types. The middle sequences were obtained from Proteus simulations without any bias ( $\delta = 0$ ). For each bias level, five low energy sequences are shown. Hydrophobic positions are colored according to the simulation where they appear first: from brick red (top) to light green (bottom). The 14 hydrophobic core positions are indicated by red arrows. (B) 3D Tiam1 structure (stereo) with residue colors as in (A). The backbone is shown in light gray.

826 amino acid positions in the Tiam1 PDZ domain known to  
 827 contribute to peptide binding specificity. Modifying these four  
 828 positions (quadruple mutant or QM) in the protein altered the  
 829 binding specificity such that QM preferentially bound a Caspr4  
 830 peptide relative to a syndecan-1 (Sdc1) peptide.<sup>70,73</sup> The four  
 831 mutations in the QM PDZ domain were L911M, K912E,  
 832 L915F, and L920 V. All four positions but Lys912 are part of  
 833 the conserved hydrophobic core. The four single and two  
 834 double mutants were also characterized experimentally.<sup>70,73</sup> For  
 835 simplicity, we denote the native (WT) sequence as LKLL and  
 836 the quadruple mutant (QM) as MEFV. Other variants are  
 837 denoted similarly. Replica Exchange MC simulations were  
 838 conducted on several structural templates, where all four  
 839 positions could mutate simultaneously, into all amino acid types  
 840 except Gly and Pro. We used the Proteus model with the lower  
 841 dielectric constant,  $\epsilon_p = 4$ , which gave similarity scores  
 842 equivalent to the  $\epsilon_p = 8$  model but had a reduced tendency  
 843 to bury polar side chains, thanks to its lower dielectric constant.  
 844 In addition, no bias energy term was used, only the MMGBSA  
 845 energy function. The CPD model used either the wildtype or  
 846 the quadruple mutant crystal structure as backbone template  
 847 for the design (PDB codes 4GVD and 4NXQ, respectively),  
 848 shown in Figure 8. Although these two structures were

determined with the Sdc1 and Caspr4 ligands, they were used here for both holo *and* apo design simulations. The backbone rms deviation between these structures is 0.9 Å, with the main differences in the flexible 850–857 loop near the peptide C-terminus and in helix  $\alpha_2$ . This helix is pushed slightly outward in the mutant complex, to accommodate Phe side chains both at protein position 915 and at the peptide C-terminus. One expectation is that the mutant backbone model (4NXQ) will better describe variants with Phe at position 915 and the wildtype backbone model (4GVD) will better describe variants with a smaller 915 side chain.

We studied six systems: the Tiam1 PDZ domain with either its wildtype or QM backbone X-ray structure, with the syndecan1 or the Caspr4 peptide ligand or no ligand. Results are shown in Figure 8. For all six systems, the native or native-like amino acid types were sampled at all four designed positions. For example, using the wildtype backbone structure (4GVD), Leu911 was preserved in the apo simulations and changed to Ile or Val in the holo simulations. Similarly, holo simulations with the mutant backbone structure (4NXQ) sampled Ile, Leu, and Met. With the mutant backbone, holo simulations sampled somewhat different types at position 911 (Trp, Arg, Lys), which all appear in low amounts at the



**Figure 8.** Design of four Tiam1 specificity positions. (A) X-ray structures (stereo) with the wildtype sequence (LKLL; labeled WT; PDB code 4GVD) or the quadruple mutant sequence (MEFV; labeled QM; PDB code 4NXQ), with bound Sdc1 and Caspr4, respectively. The four designed side chains are shown and labeled (both WT and QM mutant types). (B) Logo representation of designed sequences with no ligand (apo) or Sdc1 or Caspr4, using the wildtype (left) or quadruple mutant (right) X-ray structure. (C) Covariance plots for the 911–912 pair: populations of each pair of types are shown as levels of color, with yellow the most highly populated (5%) and red the lowest (0%). Results correspond to design simulations with the wildtype backbone.

corresponding position in the Pfam seed alignment. All the simulations sampled mostly Arg, Lys, Gln and occasionally Glu at position 912, and mostly Leu and Val at position 920, similar to the wildtype sequence. Not surprisingly, agreement with the wildtype sequence was better with the wildtype X-ray structure, while agreement with the mutant sequence was better with the mutant X-ray structure.

Recovery of the precise native and quadruple mutant sequences in the different states (apo and holo) was qualitative. Thus, using the wildtype backbone structure and in the apo state, the MC simulation recovered the wildtype sequence LKLL just 2 kcal/mol above the lowest energy sequence

(KKLV). The LKML homologue was the second best sequence overall, and the homologues IKLL and LKLV were just 1–2 kcal/mol higher in energy. The mutant sequence MEFV did not appear, nor did any close homologues, probably because the wildtype backbone structure cannot accommodate Phe at position 915. Similar results were obtained with the Sdc1 ligand, with the LKLL, IKLL, VKLL, and MKLL sequences all having energies just 1–2 kcal/mol above the best sequence. The MKLL:Sdc1 affinity is known experimentally, and is within 0.1 kcal/mol of the wildtype value.<sup>73</sup> Experimentally, the wildtype and mutant sequences have the same binding free energy for Caspr4, and stabilities just 2 kcal/mol apart,

896 suggesting that both should be sampled. Instead, neither was  
897 sampled. The closest homologue sequence was IEAV (similar  
898 to MEFV), at +2 kcal/mol (relative to the best sequence). This  
899 was probably due to steric conflict between position 915 (L or  
900 F) and the Caspr4 Phe0 in this backbone geometry.

901 Using the mutant backbone structure (4NXQ) and in the  
902 apo state, the room temperature Monte Carlo replica recovered  
903 the mutant sequence MEFV at an energy of +5 kcal/mol  
904 (relative to the best sequence) and the wildtype sequence  
905 LKLL at +7 kcal/mol. Both protein variants are thermodynamically  
906 stable; a slightly higher energy to produce LKLL seems  
907 reasonable, since the design simulation used the mutant  
908 backbone structure, which presumably should favor MEFV.  
909 With the Sdc1 ligand, MEFV appeared at an energy of +6 kcal/  
910 mol, relative to the best sequence, which was the close wildtype  
911 homologue IKLV. VKVL was just 3 kcal/mol higher. With the  
912 Caspr4 ligand, the mutant sequence appeared at an energy of  
913 +7 kcal/mol, compared to the best sequence, TKMV. Its  
914 homologues MQMV and MEMV appeared at +5 kcal/mol.  
915 The wildtype LKLL and its close homologues did not appear  
916 (indicating poorer energies), while MAFI was the second best  
917 sequence overall.

918 A more detailed comparison is possible with the binding  
919 affinities of the experimentally characterized mutants.<sup>73</sup> The  
920 experiments show that (1) affinity changes associated with each  
921 position are roughly independent of the other positions  
922 (coupling free energies of 0.4 kcal/mol or less between  
923 positions); (2) homologous changes to Leu911, Leu915, and  
924 Val920 have a very small effect on the affinity; (3) changing  
925 Lys912 to Glu reduces binding by about 0.5–1 kcal/mol (for  
926 both peptides, possibly due to lost interactions with the Lys  
927 methylenes); (4) changing Leu915 to Phe affects binding  
928 differently depending on the residue type at position 912 type  
929 and the peptide. These properties are mostly reproduced by our  
930 simulations. With the wildtype backbone model, considering  
931 sequences of the form NKNN (where N ∈ {I,L,V,M}), the  
932 mean apo and Sdc1-bound energies are  $0.9 \pm 0.6$  and  $1.1 \pm 0.5$   
933 kcal/mol, respectively, which leads to a mean affinity of  $0.2 \pm$   
934 0.8 kcal/mol (relative to IKLL, taken as a reference): mutations  
935 between the amino acid types I, L, V, and M (N to N'  
936 mutations) change the Sdc1 affinity very little, consistent with  
937 experiment. Comparing the apo and holo energies sampled in  
938 our design simulations, we predict that NKNN → NENN  
939 mutations lead to affinity changes of +0.75 kcal/mol for both  
940 peptides, compared to 0.94 kcal/mol (Sdc1) and 0.55 kcal/mol  
941 (Caspr4) experimentally. Similarly, we predict that NKNN →  
942 NKFN mutations reduce the affinity by 0.5 kcal/mol for both  
943 peptides, compared to 1.2 and 0.8 kcal/mol experimentally.  
944 Only for NENN → NEFN mutations do we see larger errors:  
945 we predict a 0.5 kcal/mol affinity loss for Sdc1 (vs no loss  
946 experimentally) and a 0.9 kcal/mol loss for Caspr4 (vs a 0.5  
947 kcal/mol gain experimentally). Specificity changes are predicted  
948 to be small, in qualitative agreement with experiment. For  
949 example the MKFV → MEFV mutation favors Caspr4, relative  
950 to Sdc1, by 0.2 kcal/mol, compared to 0.5 kcal/mol  
951 experimentally for the homologous LKLL → LELL mutation.  
952 The simulations also gave information on correlations  
953 between the four mutating positions. Figure 8C shows  
954 covariance plots between positions 911 and 912 for the apo  
955 and holo simulations. Position 911 was more diverse in the apo  
956 than in either holo state (Sdc1 or Caspr4), while 912 was not  
957 very sensitive to the peptide. The computed pairwise  
958 correlations among all four protein positions were weak, so

that the covariance plots mostly exhibit horizontal and vertical  
959 lines or bands, without noticeable “diagonal” features. This  
960 agrees with the experimental affinities of the single, double, and  
961 quadruple mutants, where the affinity changes associated with  
962 each point mutation were only weakly coupled to the other  
963 positions.<sup>73</sup>

## 5. DISCUSSION

964 **5.1. Model Limitations.** We have parametrized a simple  
965 CPD model for PDZ design, suitable for high-throughput  
966 design applications and implemented in the Proteus software.  
967 For the folded state representation, we use a high-quality  
968 protein force field and Generalized Born solvent model. We  
969 tested two protein dielectric constants  $\epsilon_p$ . We used a specific set  
970 of X-ray structures for our test proteins, each with a specific  
971 backbone conformation. For the side chains, we used a simple,  
972 discrete rotamer library and a short minimization of each side  
973 chain pair during the energy matrix calculation to alleviate the  
974 discrete rotamer approximation. Both the energy function and  
975 the rotamer description have been extensively tested and shown  
976 to give very good performance for side chain reconstruction  
977 tests<sup>54</sup> (comparable to the popular Scwrl4 program<sup>53</sup>) and  
978 good performance for a large set of protein acid/base  
979 constants<sup>74</sup> (superior to the Rosetta software,<sup>75</sup> despite its  
980 extensive *ad hoc* parameter tuning).

981 The unfolded state representation used a simple, implicit  
982 model, characterized by a set of empirical, amino acid chemical  
983 potentials or reference energies. These energies were chosen by  
984 a likelihood maximization procedure, formulated here, in order  
985 to reproduce the amino acid composition of carefully selected  
986 natural homologues. The unfolded state description used here  
987 is more refined than previously,<sup>76</sup> since distinct reference  
988 energy values were used for amino acid positions that are  
989 buried or exposed in the *folded* state. This method assumes that  
990 there is residual structure in the unfolded state, with some  
991 positions more buried than others. Furthermore, it should make  
992 the parametrization more robust and less sensitive to the size  
993 and structure of the natural homologues used to define the  
994 target amino acid compositions, because the amino acid  
995 frequencies of exposed and buried regions are averaged  
996 separately. In principle, this doubles the number of adjustable  
997 reference energies. However, we reduced this number by  
998 introducing amino acid similarity classes, with one adjustable  
999 reference energy per class. To optimize the reference energies,  
1000 we performed design calculations for each test protein (apo  
1001 state) where half of the amino positions could mutate at a time  
1002 (excluding Gly and Pro), with distinct simulations for each half.  
1003 This way, during parameter optimization, a mutating position  
1004 was always surrounded by an environment at least 50%  
1005 identical to the wildtype sequence. The design calculations  
1006 relied on a powerful and efficient Replica Exchange Monte  
1007 Carlo exploration method that used over a half billion steps per  
1008 simulation (per replica), and produced thousands of designed  
1009 sequences in a single simulation. Reference energy values were  
1010 optimized with two different choices for the protein dielectric  
1011 constant  $\epsilon_p$ . The performance levels were similar for both  
1012 values.

1013 The model has several limitations, most of which are  
1014 widespread in CPD implementations and applications. The first  
1015 is the use of protein stability as the sole design criterion,  
1016 without explicitly accounting for fold specificity,<sup>77</sup> protection  
1017 against aggregation, or functional considerations like ligand  
1018 binding. We note, however, that the Superfamily tests did not  
1019

lead to any fold misassignments (sequences that prefer another SCOP fold), so that in practice, fold specificity was achieved. Functional criteria can also be introduced in an *ad hoc* way; for example, the sequences tested by MD were designed with 11 peptide binding residues fixed, to facilitate future experimental studies.

A second model limitation is the use of a fixed protein backbone. In fact, the backbone is not really fixed: rather, certain motions are allowed but modeled *implicitly*, through the use of a protein dielectric constant greater than 1 ( $\epsilon_p = 4$  or 8).<sup>78</sup> This dielectric value means that the protein structure (including its backbone) is allowed to relax or reorganize in response to charge redistribution associated with mutations or side chain rotamer changes. However, the reorganization is modeled implicitly, not explicitly,<sup>78</sup> and it does not involve motion of the atomic centers or their associated van der Waals spheres. Thus, the backbone cannot reorganize in response to steric repulsion produced by mutations or rotamer changes, nor can it shift to fill space left empty by a mutation. The effect of this approximation was apparent in the design of the four Tiam1 specificity positions, where the designed sequences were sensitive to the particular backbone conformation of the protein and peptide. Specifically, with the wildtype backbone structure, there was no room to insert a Phe side chain at position 915, even though Phe915 is present in the experimental quadruple mutant (which has a slightly different backbone structure). Therefore, the choice of the initial X-ray structural model is important, and several strategies are possible. Here, to parametrize the CPD model, we used X-ray structures solved with a peptide ligand, even though the parametrization simulations and most of the testing were done for the apo proteins. This choice was made partly because the apo/holo PDZ structures are quite similar and partly to make the model more transferable and facilitate applications to peptide binding. Another strategy could have been to parametrize the model using all apo structures, then switch to holo structures for the Tiam1 application.

For whole protein design (such as the hydrophobic titration application), the use of a fixed backbone can be partly counterbalanced by designing two or more PDZ structures. For example, pooling the designed Tiam1 and Cask sequences gave a mean sequence entropy comparable to the experimental Pfam set, and allowed us to recapitulate more sequences than design with just one backbone. In the application to Tiam1 4-position design, the fixed backbone was also counterbalanced by doing calculations separately with two different backbone structures, a holo wildtype and a holo mutant structure. Simulations with the mutant backbone allowed us to obtain mutants having Phe at position 915. Notice that a new method for multibackbone design was recently developed in Proteus, based on a novel, nonheuristic hybrid Monte Carlo method that preserves Boltzmann sampling.<sup>35</sup> This method could be applied in the future.

A third limitation of our model is the need, for optimal results, to parametrize the reference energies specifically for a given set of proteins. This step is well-automated and highly parallel. However, it involves several choices that are partly arbitrary. These include the choice of a set of protein domains to represent the protein or family of interest. We also need to choose a similarity threshold to define the target homologues from which we compute the experimental amino acid compositions. Here, we chose to use close homologues of each family member, compute their compositions, then average

over the two family representatives. This method worked well but other choices are possible, and more work is needed to draw definitive conclusions. Also, the monoposition design of Tiam1 showed evidence of some systematic error (Figure 4), with a large fraction of Arg, Lys, and Gln residues types on the protein surface, despite the optimized reference energies. In the future, it may be necessary to relax the intragroup constraints toward the end of the reference energy optimization and/or target smaller numbers of mutating positions, instead of one-half of the protein at a time.

A fourth limitation of our model is the discrete rotamer approximation, which requires some adaptation of the energy function to avoid exaggerated steric clashes; the method used here is the residue-pair minimization method described earlier.<sup>26,76</sup> A fifth limitation is the use of a pairwise additive solvation model (as in most CPD models). Specifically, the dielectric environment of each residue pair is assumed here to be native-like (so-called “Native Environment approximation” or NEA<sup>74,76</sup>). This leads to an energy function that has the form of a sum over pairs of residues and that can be precalculated and stored in an energy matrix, which then serves as a lookup table during the subsequent Monte Carlo simulations. Despite this approximation, the model gave good results for a large acid/base benchmark, a problem that is very sensitive to the electrostatic treatment.<sup>74</sup>

Some of these limitations could be removed in future work. In particular, since the energy function is mostly physics-based, it can benefit rapidly from ongoing improvements in protein force fields and solvation models. Thus, the NEA approximation could be removed in the future due to the recent implementation (manuscript in preparation) of a more exact Generalized Born calculation, whose efficiency is comparable to the pairwise approximation.<sup>79</sup> We have also implemented an improved model for hydrophobic solvation,<sup>80</sup> which is faster and more accurate than our current surface area energy term (manuscript in preparation).

**5.2. Model Testing and Applications.** Designed sequences were extensively compared to natural sequences, through fold recognition tests, similarity calculations, and entropy calculations. In the test simulations, we designed the entire protein sequence, so that all positions (except Gly and Pro) could mutate freely, subject only to an overall bias toward the mean, experimental amino acid composition (through the reference energies). Despite the lack of experimental bias or constraints, the resulting sequences had a high overall similarity to the natural, Pfam sequences, as measured by the Blosum40 similarity scores. The scores obtained were mostly comparable to the similarity scores between pairs of Pfam sequences themselves. Thus, the sequences designed with Proteus resemble moderately distant natural homologues. The similarity was very strong for residues in the core of the protein, as observed in previous CPD studies.<sup>30,72</sup> In contrast, for residues at the protein surface, similarity scores were close to zero, the score one would obtain if one picked amino acid types randomly. Notice that many surface residues are involved in functional interactions, such as the 11 peptide-binding residues in PDZ domains. Surface residues are also selected by evolution to avoid aggregation or unwanted adhesion. Most of these functional constraints are not explicitly accounted for in our design protocol (although the energy function indirectly favors protein solubility). Despite the limited similarity scores for surface regions, fold recognition with the Superfamily tool and the best design models was almost perfect. Earlier fold

recognition tests that used a simpler energy function gave a lower fold recognition rate of about 85% (for a larger and more diverse test set) and lower similarities.<sup>15,50</sup> Evidently, the combined use of an improved protein force field, Generalized Born solvent, and family specific reference energies leads to designed sequences that are more native-like and presumably better.

The Proteus sequences were also compared to sequences designed with the Rosetta software (using default parameters), which has itself been extensively tested. On the basis of the Blosum similarity scores (vs natural sequences in Pfam) and the fold recognition tests, the Proteus and Rosetta sequences appear to be of about the same quality. However, Rosetta makes fewer mutations than Proteus, so that the identity scores, compared to the corresponding wildtype protein, are about 6% points higher. This means that Proteus mutates about five more positions, on average, per PDZ domain. This number could easily be reduced, by adding to the Proteus energy function an explicit bias energy term that increases with the number of mutations (away from the wildtype sequence). An equivalent bias energy was used above for just two simulations of Tiam1 and Cask (see the “biased Proteus” results in the two upper panels of Figure 5), to illustrate the possibility of using experimentally restrained sampling. It remains to be seen whether a restraint based on the identity score would lead to more stable and realistic designed sequences.

Another attractive route for testing designed sequences is through high-level MD simulations. Here, 10 designed Tiam1 sequences were tested in rather long MD simulations, in the apo form, using the same protein force field as in the CPD model (Amber force field) and an established explicit water model. These sequences were designed using Proteus, with Gly, Pro, and 11 peptide-binding positions held fixed but all others free to mutate. Six of the sequences remained stable over 200 ns simulation lengths and two were extended up to a microsecond, which represents a very stringent test of the designs. Sequence 6 was stable over the whole microsecond. The mean deviation of seq-6 from the starting, experimental wildtype structure was 1 Å, which is the same as the mean deviation in the MD simulation of the wildtype sequence itself. The deviation between the mean sequence 6 MD structure and the mean wildtype MD structure was also small, just 1.2 Å. Sequence 2 was also simulated and remained stable until just before the end of the microsecond simulation, at which point it underwent a larger fluctuation. The fluctuation regressed 100 ns later. An even longer simulation would be needed to determine if this fluctuation is harmless or signals the beginning of domain unfolding. Note that in the presence of a peptide ligand, we expect the structural stability of the designed domains would increase further. MD simulations of additional designed sequences would also be of interest. Direct experimental testing of the designed proteins remains to be done.

The CPD model was used for two applications. “Hydrophobic titration” of two PDZ domains illustrated a novel way to characterize protein designability. The cost or availability of hydrophobic side chain types was controlled by a bias energy term that was gradually varied. The mean overall hydrophobic “susceptibility”, the number of type of changes per kcal/mol and per position, differed by about 20% between Tiam1 and Cask. In Tiam1, 11 of the core positions remained hydrophobic even with the largest bias value favoring polar types, while 14 other positions changed to nonpolar types at the largest

nonpolar bias energy value. A comparison to other domain families would be of interest, and is left for future work.

Redesign of four specificity positions in Tiam1 allowed us to test the design model in a different way. It revealed some of the limitations of fixed backbone design, but also gave semi-quantitative agreement with available binding free energies. This agreement predicts new mutations that could be of interest for obtaining new specificity properties. They remain to be studied further and tested experimentally. Here, the apo and two holo states were studied, and designed separately. Information about binding affinities and binding specificity were then obtained by comparing the energies sampled in the different simulations. In the future, we would like to include binding affinity and/or specificity directly in the design calculations, as a property to be designed for or against within a single simulation. In addition, we should allow different backbone structures for the apo and each holo system. This could be done in the future, since recent hybrid Monte Carlo schemes<sup>35,81</sup> can be used for multibackbone design, and can be extended to the problem of designing ligand binding specificity. We also note that since our energy function is physics-based, it has transferability to a range of molecule types, such as nonnatural amino acids (considered in an earlier protein–ligand design study<sup>34</sup>). Such amino acids could be of interest for designing PDZ peptide ligands, to provide additional diversity and perhaps enhanced resistance to proteolysis.

## ASSOCIATED CONTENT

### Supporting Information

The Supporting Information is available free of charge on the ACS Publications website at DOI: [10.1021/acs.jctc.6b01255](https://doi.org/10.1021/acs.jctc.6b01255).

Detailed description of the model cross validation and results obtained with a protein dielectric  $\epsilon_p = 4$  ([PDF](#))

## AUTHOR INFORMATION

### Corresponding Author

\*E-mail: [thomas.simonson@polytechnique.fr](mailto:thomas.simonson@polytechnique.fr).

### ORCID

Nicolas Panel: [0000-0001-8782-0586](https://orcid.org/0000-0001-8782-0586)

Thomas Simonson: [0000-0002-5117-7338](https://orcid.org/0000-0002-5117-7338)

### Author Contributions

<sup>§</sup>These authors contributed equally to the manuscript.

### Notes

The authors declare no competing financial interest.

## ACKNOWLEDGMENTS

We are grateful for discussions with Michael Schnieders and Young Joo Sun (University of Iowa). Some of the calculations were done at the CINES supercomputer center of the French Ministry of Research. NAMD was developed by the Theoretical and Computational Biophysics Group in the Beckman Institute at the University of Illinois at Urbana.

## REFERENCES

- (1) Harris, B. Z.; Lim, W. A. Mechanism and role of PDZ domains in signaling complex assembly. *J. Cell Sci.* **2001**, *114*, 3219–3231.
- (2) Hung, A. Y.; Sheng, M. PDZ Domains: Structural Modules for Protein Complex Assembly. *J. Biol. Chem.* **2002**, *277*, 5699–5702.
- (3) Tonikian, R.; Zhang, Y. N.; Sazinsky, S. L.; Currell, B.; Yeh, J. H.; Reva, B.; Held, H. A.; Appleton, B. A.; Evangelista, M.; Wu, Y.; Xin, X. F.; Chan, A. C.; Seshagiri, S.; Lasky, L. A.; Sander, C.; Boone, C.;

- 1266 Bader, G. D.; Sidhu, S. S. A Specificity Map for the PDZ Domain  
1267 Family. *PLoS Biol.* **2008**, *6*, 2043–2059.  
(4) Gfeller, D.; Butty, F.; Wierzbicka, M.; Verschueren, E.; Vanhee,  
1268 P.; Huang, H.; Ernst, A.; Darand, N.; Stagljar, I.; Serrano, L.; Sidhu, S.  
1269 S.; Bader, G. D.; Kim, P. M. The multiple-specificity landscape of  
1270 modular peptide recognition domains. *Mol. Syst. Biol.* **2011**, *7*, art484.  
(5) Subbiah, V. K.; Kranjec, C.; Thomas, M.; Banks, L. PDZ  
1271 domains: the building blocks regulating tumorigenesis. *Biochem. J.*  
1272 **2011**, *439*, 195–205.  
(6) Shepherd, T. R.; Fuentes, E. J. Structural and thermodynamic  
1273 analysis of PDZ-ligand interactions. *Methods Enzymol.* **2011**, *488*, 81–  
1277 100.  
(7) Bacha, A.; Clausen, B. H.; Moller, M.; Vestergaard, B.; Chic, C.  
1278 N.; Round, A.; Sørensen, P. L.; Nissen, K. B.; Kastrup, J. S.; Gajhede,  
1279 M.; Jemth, P.; Kristensen, A. S.; Lundstrom, P.; Lambertsen, K. L.;  
1280 Stromgaard, K. A high-affinity, dimeric inhibitor of PSD-95 bivalently  
1281 interacts with PDZ1–2 and protects against ischemic brain damage.  
1282 *Proc. Natl. Acad. Sci. U. S. A.* **2012**, *109*, 3317–3322.  
(8) Roberts, K. E.; Cushing, P. R.; Boisguerin, P.; Madden, D. R.;  
1283 Donald, B. R. Computational Design of a PDZ Domain Peptide  
1284 Inhibitor that Rescues CFTR Activity. *PLoS Comput. Biol.* **2012**, *8*,  
1285 e1002477.  
(9) Zheng, F.; Jewell, H.; Fitzpatrick, J.; Zhang, J.; Mierke, D. F.;  
1286 Grigoryan, G. Computational Design of Selective Peptides to  
1287 Discriminate between Similar PDZ Domains in an Oncogenic  
1288 Pathway. *J. Mol. Biol.* **2015**, *427*, 491–510.  
(10) Lockless, W.; Ranganathan, R. Evolutionary Conserved  
1289 Pathways of Energetic Connectivity in Protein Families. *Science*  
1290 **1999**, *285*, 295–299.  
(11) Kong, Y.; Karplus, M. Signaling pathways of PDZ2 domain: A  
1291 molecular dynamics interaction correlation analysis. *Proteins: Struct.,  
1292 Funct., Genet.* **2009**, *74*, 145–154.  
(12) McLaughlin, R. N., Jr; Poelwijk, F. J.; Raman, A.; Gosal, W. S.;  
1293 Ranganathan, R. The spatial architecture of protein function and  
1294 adaptation. *Nature* **2012**, *491*, 138–142.  
(13) Melero, C.; Ollikainen, N.; Harwood, I.; Karpiai, J.; Kortemme,  
1295 T. Quantification of the transferability of a designed protein specificity  
1296 switch reveals extensive epistasis in molecular recognition. *Proc. Natl.  
1297 Acad. Sci. U. S. A.* **2014**, *111*, 15426–15431.  
(14) Reina, J.; Lacroix, E.; Hobson, S. D.; Fernandez-Ballester, G.;  
1298 Rybin, V.; Schwab, M. S.; Serrano, L.; Gonzalez, C. Computer-aided  
1299 design of a PDZ domain to recognize new target sequences. *Nat.  
1300 Struct. Biol.* **2002**, *9*, 621–627.  
(15) Schmidt am Busch, M.; Sedano, A.; Simonson, T. Computational  
1301 protein design: validation and possible relevance as a tool for  
1302 homology searching and fold recognition. *PLoS One* **2010**, *5* (5),  
1303 e10410.  
(16) Smith, C. A.; Kortemme, T. Structure-Based Prediction of the  
1304 Peptide Sequence Space Recognized by Natural and Synthetic PDZ  
1305 Domains. *J. Mol. Biol.* **2010**, *402*, 460–474.  
(17) Butterfoss, G. L.; Kuhlman, B. Computer-based design of novel  
1306 protein structures. *Annu. Rev. Biophys. Biomol. Struct.* **2006**, *35*, 49–65.  
(18) Lippow, S. M.; Tidor, B. Progress in computational Protein  
1307 Design. *Curr. Opin. Biotechnol.* **2007**, *18*, 305–311.  
(19) Dai, L.; Yang, Y.; Kim, H. R.; Zhou, Y. Improving computational  
1308 protein design by using structure-derived sequence profile. *Proteins:  
1309 Struct., Funct., Genet.* **2010**, *78*, 2338–2348.  
(20) Feldmeier, K.; Hoecker, B. Computational protein design of  
1310 ligand binding and catalysis. *Curr. Opin. Chem. Biol.* **2013**, *17*, 929–  
1311 933.  
(21) Tinberg, C. E.; Khare, S. D.; Dou, J.; Doyle, L.; Nelson, J. W.;  
1312 Schena, A.; Jankowski, W.; Kalodimos, C. G.; Johnsson, K.; Stoddard,  
1313 B. L.; Baker, D. Computational design of ligand-binding proteins with  
1314 high affinity and selectivity. *Nature* **2013**, *501*, 212–218.  
(22) Au, L.; Green, D. F. Direct Calculation of Protein Fitness  
1315 Landscapes through Computational Protein Design. *Biophys. J.* **2016**,  
1316 *110*, 75–84.  
(23) Pokala, N.; Handel, T. Energy functions for protein design I: Efficient and accurate continuum electrostatics and solvation. *Protein Sci.* **2004**, *13*, 925–936.  
(24) Samish, I.; MacDermaid, C. M.; Perez-Aguilar, J. M.; Saven, J. G. Theoretical and computational protein design. *Annu. Rev. Phys. Chem.* **2011**, *62*, 129–149.  
(25) Li, L.; Francklyn, C.; Carter, C. W. Aminoacylating Urzymes Challenge the RNA World Hypothesis. *J. Biol. Chem.* **2013**, *288*, 26856–26863.  
(26) Schmidt am Busch, M.; Lopes, A.; Mignon, D.; Simonson, T. Computational protein design: software implementation, parameter optimization, and performance of a simple model. *J. Comput. Chem.* **2008**, *29*, 1092–1102.  
(27) Simonson, T. Protein:ligand recognition: simple models for electrostatic effects. *Curr. Pharm. Des.* **2013**, *19*, 4241–4256.  
(28) Polydorides, S.; Michael, E.; Mignon, D.; Druart, K.; Archontis, G.; Simonson, T. In *Methods in Molecular Biology: Design and Creation of Protein Ligand Binding Proteins*; Stoddard, B., Ed.; Springer Verlag: New York, 2016; Vol. 1414; p 0000.  
(29) Kuhlman, B.; Baker, D. Native protein sequences are close to optimal for their structures. *Proc. Natl. Acad. Sci. U. S. A.* **2000**, *97*, 10383–10388.  
(30) Dantas, G.; Kuhlman, B.; Callender, D.; Wong, M.; Baker, D. A Large Test of Computational Protein Design: Folding and Stability of Nine Completely Redesigned Globular Proteins. *J. Mol. Biol.* **2003**, *332*, 449–460.  
(31) Rohl, C. A.; Strauss, C. E. M.; S, M. K. M.; Baker, D. Protein structure prediction using Rosetta. *Methods Enzymol.* **2004**, *383*, 10383–103866–93.  
(32) Lazaridis, T.; Karplus, M. Effective energy function for proteins in solution. *Proteins: Struct., Funct., Genet.* **1999**, *35*, 133–152.  
(33) Mignon, D.; Simonson, T. Comparing three stochastic search algorithms for computational protein design: Monte Carlo, Replica Exchange Monte Carlo, and a multistart, steepest-descent heuristic. *J. Comput. Chem.* **2016**, *37*, 1781–1793.  
(34) Druart, K.; Palmai, Z.; Omarjee, E.; Simonson, T. Protein:ligand binding free energies: a stringent test for computational protein design. *J. Comput. Chem.* **2016**, *37*, 404–415.  
(35) Druart, K.; Bigot, J.; Audit, E.; Simonson, T. A hybrid Monte Carlo method for multibackbone protein design. *J. Chem. Theory Comput.* **2016**, *12*, 6035–6048.  
(36) Gaillard, T.; Simonson, T. Full protein sequence redesign with an MMGBSA energy function. *J. Comput. Chem.* **2017**.  
(37) Baker, D. Prediction and design of macromolecular structures and interactions. *Philos. Trans. R. Soc., B* **2006**, *361*, 459–463.  
(38) Frenkel, D.; Smit, B. *Understanding molecular simulation*; Academic Press: New York, 1996; Chapter 3.  
(39) Grimmett, G. R.; Stirzaker, D. R. *Probability and random processes*; Oxford University Press: Oxford, United Kingdom, 2001.  
(40) Kleinman, C. L.; Rodrigue, N.; Bonnard, C.; Philippe, H.; Lartillot, N. A maximum likelihood framework for protein design. *BMC Bioinf.* **2006**, *7*, Art326.  
(41) Fowler, R. H.; Guggenheim, E. A. *Statistical Thermodynamics*; Cambridge University Press: Cambridge, United Kingdom, 1939.  
(42) Cornell, W.; Cieplak, P.; Bayly, C.; Gould, I.; Merz, K.; Ferguson, D.; Spellmeyer, D.; Fox, T.; Caldwell, J.; Kollman, P. A. Second Generation Force Field for the Simulation of Proteins, Nucleic Acids, and Organic Molecules. *J. Am. Chem. Soc.* **1995**, *117*, 5179–5197.  
(43) Aleksandrov, A.; Polydorides, S.; Archontis, G.; Simonson, T. Predicting the Acid/Base Behavior of Proteins: A Constant-pH Monte Carlo Approach with Generalized Born Solvent. *J. Phys. Chem. B* **2010**, *114*, 10634–10648.  
(44) Hawkins, G. D.; Cramer, C.; Truhlar, D. Pairwise descreening of solute charges from a dielectric medium. *Chem. Phys. Lett.* **1995**, *246*, 122–129.  
(45) Lopes, A.; Aleksandrov, A.; Bathelt, C.; Archontis, G.; Simonson, T. Computational sidechain placement and protein mutagenesis with

- 1401 implicit solvent models. *Proteins: Struct., Funct., Genet.* **2007**, *67*, 853–  
1402 867.
- 1403 (46) Lee, B.; Richards, F. The interpretation of protein structures:  
1404 estimation of static accessibility. *J. Mol. Biol.* **1971**, *55*, 379–400.
- 1405 (47) Brünger, A. T. *X-PLOR version 3.1, A System for X-ray*  
1406 *crystallography and NMR*; Yale University Press: New Haven, 1992.
- 1407 (48) Gaillard, T.; Simonson, T. Pairwise Decomposition of an  
1408 MMGBSA Energy Function for Computational Protein Design. *J.*  
1409 *Comput. Chem.* **2014**, *35*, 1371–1387.
- 1410 (49) Street, A. G.; Mayo, S. Pairwise calculation of protein solvent-  
1411 accessible surface areas. *Folding Des.* **1998**, *3*, 253–258.
- 1412 (50) Schmidt am Busch, M.; Mignon, D.; Simonson, T. Computational  
1413 protein design as a tool for fold recognition. *Proteins: Struct.,*  
1414 *Funct., Genet.* **2009**, *77*, 139–158.
- 1415 (51) Eswar, N.; Marti-Renom, M. A.; Webb, B.; Madhusudhan, M. S.;  
1416 Eramian, D.; Shen, M.; Pieper, U.; Sali, A. Comparative Protein  
1417 Structure Modeling With MODELLER. *Curr. Prot. Bioinf.* **2006**, *Suppl.*  
1418 *15*, S.6.1–S.6.30.
- 1419 (52) Tuffery, P.; Etchebest, C.; Hazout, S.; Lavery, R. A New  
1420 Approach to the Rapid Determination of Protein Side Chain  
1421 Conformations. *J. Biomol. Struct. Dyn.* **1991**, *8*, 1267–1289.
- 1422 (53) Krivov, G. G.; Shapalov, M. V.; Dunbrack, R. L. Improved  
1423 prediction of protein side-chain conformations with SCWRL4.  
1424 *Proteins: Struct., Funct., Genet.* **2009**, *77*, 778–795.
- 1425 (54) Gaillard, T.; Panel, N.; Simonson, T. Protein sidechain  
1426 conformation predictions with an MMGBSA energy function. *Proteins:*  
1427 *Struct., Funct., Genet.* **2016**, *84*, 803–819.
- 1428 (55) Kofke, D. A. On the acceptance probability of replica-exchange  
1429 Monte Carlo trials. *J. Chem. Phys.* **2002**, *117*, 6911–6914.
- 1430 (56) Earl, D.; Deem, M. W. Parallel tempering: theory, applications,  
1431 and new perspectives. *Phys. Chem. Chem. Phys.* **2005**, *7*, 3910–3916.
- 1432 (57) Gough, J.; Karplus, K.; Hughey, R.; Chothia, C. Assignment of  
1433 homology to genome sequences using a library of hidden Markov  
1434 models that represent all proteins of known structure. *J. Mol. Biol.*  
1435 **2001**, *313*, 903–919.
- 1436 (58) Wilson, D.; Madera, M.; Vogel, C.; Chothia, C.; Gough, J. The  
1437 SUPERFAMILY database in 2007: families and functions. *Nucleic*  
1438 *Acids Res.* **2007**, *35*, D308–D313.
- 1439 (59) Andreeva, A.; Howorth, D.; Brenner, S. E.; Hubbard, J. J.;  
1440 Chothia, C.; Murzin, A. G. SCOP database in 2004: refinements  
1441 integrate structure and sequence family data. *Nucleic Acids Res.* **2004**,  
1442 *32*, D226–229.
- 1443 (60) Durbin, R.; Eddy, S. R.; Krogh, A.; Mitchison, G. *Biological*  
1444 *sequence analysis*; Cambridge University Press: Cambridge, United  
1445 Kingdom, 2002.
- 1446 (61) Murphy, L. R.; Wallqvist, A.; Levy, R. M. Simplified amino acid  
1447 alphabets for protein fold recognition and implications for folding.  
1448 *Protein Eng., Des. Sel.* **2000**, *13*, 149–152.
- 1449 (62) Launay, G.; Mendez, R.; Wodak, S. J.; Simonson, T.  
1450 Recognizing protein-protein interfaces with empirical potentials and  
1451 reduced amino acid alphabets. *BMC Bioinf.* **2007**, *8*, 270–291.
- 1452 (63) Jo, S.; Kim, T.; Iyer, V. G.; Im, W. CHARMM-GUI: a web-  
1453 based graphical user interface for CHARMM. *J. Comput. Chem.* **2008**,  
1454 *29*, 1859–1865.
- 1455 (64) Nose, S. A unified formulation of the constant temperature  
1456 molecular dynamics method. *J. Chem. Phys.* **1984**, *81*, 511–519.
- 1457 (65) Hoover, W. G. Canonical dynamics: equilibrium phase-space  
1458 distributions. *Phys. Rev. A: At., Mol., Opt. Phys.* **1985**, *31*, 1695–1697.
- 1459 (66) Darden, T. In *Computational Biochemistry & Biophysics*; Becker,  
1460 O., Mackerell, A., Jr., Roux, B., Watanabe, M., Eds.; Marcel Dekker:  
1461 N.Y., 2001; Chapter 4.
- 1462 (67) Jorgensen, W.; Chandrasekhar, J.; Madura, J.; Impey, R.; Klein,  
1463 M. Comparison of simple potential functions for simulating liquid  
1464 water. *J. Chem. Phys.* **1983**, *79*, 926–935.
- 1465 (68) Brooks, B.; Brooks, C. L., III; Mackerell, A. D., Jr.; Nilsson, L.;  
1466 Petrella, R. J.; Roux, B.; Won, Y.; Archontis, G.; Bartels, C.; Boresch,  
1467 S.; Caflisch, A.; Caves, L.; Cui, Q.; Dinner, A. R.; Feig, M.; Fischer, S.;  
1468 Gao, J.; Hodoscek, M.; Im, W.; Kuczera, K.; Lazaridis, T.; Ma, J.;  
1469 Ovchinnikov, V.; Paci, E.; Pastor, R. W.; Post, C. B.; Pu, J. Z.; Schaefer,  
1470 York, D. M.; Karplus, M. CHARMM: The biomolecular simulation  
1471 program. *J. Comput. Chem.* **2009**, *30*, 1545–1614.
- 1472 (69) Phillips, J. C.; Braun, R.; Wang, W.; Gumbart, J.; Tajkhorshid,  
1473 E.; Villa, E.; Chipot, C.; Skeel, R. D.; Kale, L.; Schulten, K. Scalable  
1474 molecular dynamics with NAMD. *J. Comput. Chem.* **2005**, *26*, 1781–1475  
1476 1802.
- 1477 (70) Liu, X.; Speckhard, D. C.; Shepherd, T. R.; Sun, Y. J.; Hengel, S.;  
1478 Yu, L.; Fowler, C. A.; Gakhar, L.; Fuentes, E. J. Distinct roles for  
1479 conformational dynamics in protein-ligand interactions. *Structure* **2016**,  
1480 *24*, 2053–2066.
- 1481 (71) Madera, M.; Vogel, C.; Kummerfeld, S. K.; Chothia, C.; Gough,  
1482 J. The SUPERFAMILY database in 2004: additions and improvements.  
1483 *Nucleic Acids Res.* **2004**, *32*, D235–D239.
- 1484 (72) Jaramillo, A.; Wernisch, L.; Héry, S.; Wodak, S. Folding free  
1485 energy function selects native-like protein sequences in the core but  
1486 not on the surface. *Proc. Natl. Acad. Sci. U. S. A.* **2002**, *99*, 13554–1486  
1487 13559.
- 1488 (73) Shepherd, T. R.; Hard, R. L.; Murray, A. M.; Pei, D.; Fuentes, E.;  
1489 J. Distinct Ligand Specificity of the Tiam1 and Tiam2 PDZ Domains.  
1490 *Biochemistry* **2011**, *50*, 1296–1308.
- 1491 (74) Polydorides, S.; Simonson, T. Monte Carlo simulations of  
1492 proteins at constant pH with generalized Born solvent, flexible  
1493 sidechains, and an effective dielectric boundary. *J. Comput. Chem.* **2013**,  
1494 *34*, 2742–2756.
- 1495 (75) Kilambi, K.; Gray, J. J. Rapid calculation of protein pK<sub>a</sub> values  
1496 using Rosetta. *Biophys. J.* **2012**, *103*, 587–595.
- 1497 (76) Simonson, T.; Gaillard, T.; Mignon, D.; Schmidt am Busch, M.;  
1498 Lopes, A.; Amara, N.; Polydorides, S.; Sedano, A.; Archontis, K. D. G.  
1499 Computational protein design: the Proteus software and selected  
1500 applications. *J. Comput. Chem.* **2013**, *34*, 2472–2484.
- 1501 (77) Mach, P.; Koehl, P. Capturing protein sequence-structure  
1502 specificity using computational sequence design. *Proteins: Struct.,*  
1503 *Funct., Genet.* **2013**, *81*, 1556–1570.
- 1504 (78) Simonson, T. What Is the Dielectric Constant of a Protein  
1505 When Its Backbone Is Fixed? *J. Chem. Theory Comput.* **2013**, *9*, 4603–  
1506 4608.
- 1507 (79) Archontis, G.; Simonson, T. A residue-pairwise Generalized  
1508 Born scheme suitable for protein design calculations. *J. Phys. Chem. B*  
1509 **2005**, *109*, 22667–22673.
- 1510 (80) Aguilar, B.; Shadrach, R.; Onufriev, A. V. Reducing the  
1511 secondary structure bias in the generalized Born model via R6 effective  
1512 radii. *J. Chem. Theory Comput.* **2011**, *6*, 3613–3630.
- 1513 (81) Ollikainen, N.; de Jong, R. M.; Kortemme, T. Coupling Protein  
1514 Side-Chain and Backbone Flexibility Improves the Re-design of  
1515 Protein-Ligand Specificity. *PLoS Comput. Biol.* **2015**, *1*, e1004335.

#### 4.9. Application : Croissance du noyau hydrophobe

Figure 4.20 – Structure native Cask avec les hydrophobes pour des  $\delta$  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.

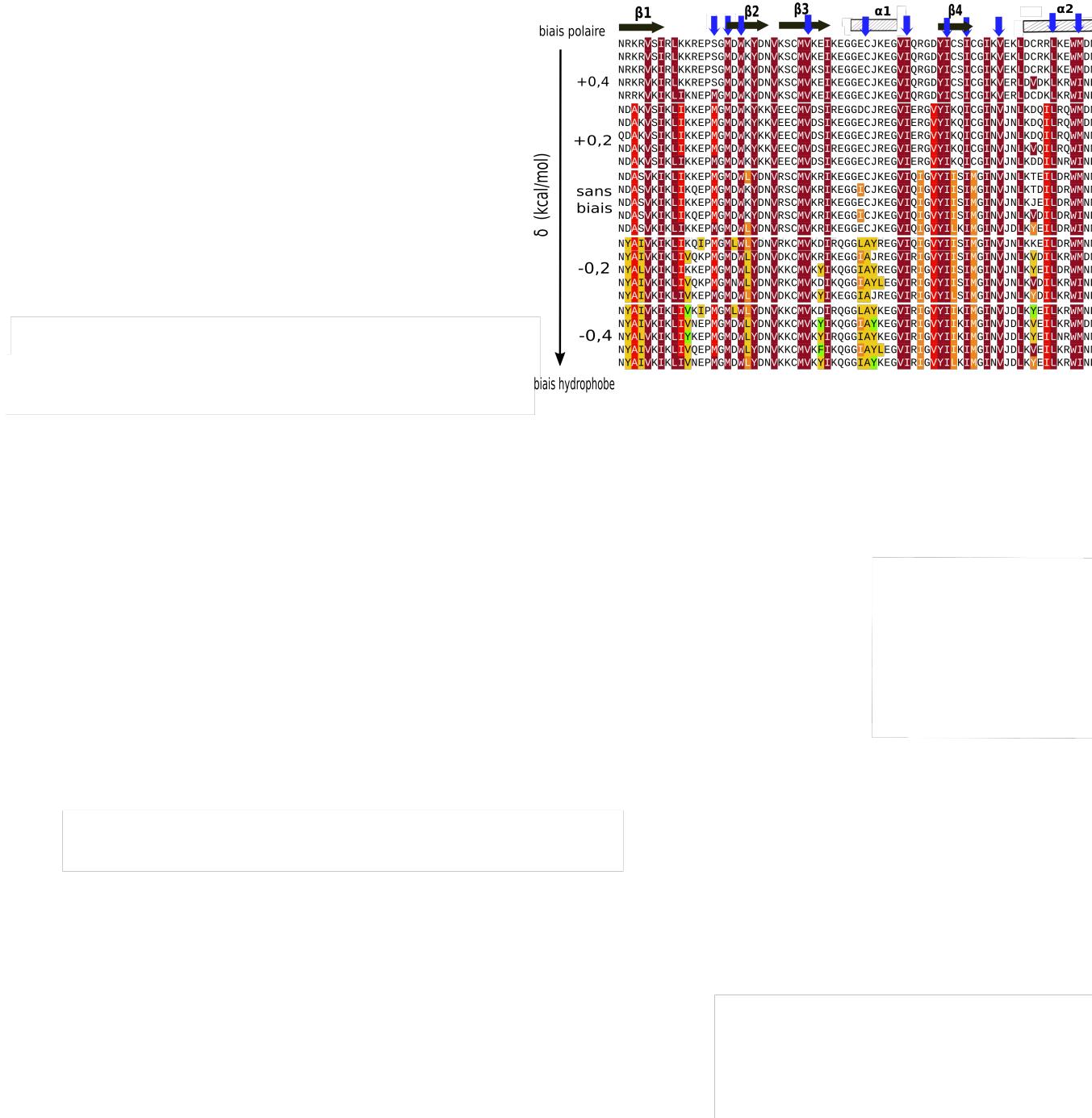


Figure 4.21 – Séquences Cask obtenues avec un delta des énergies de références à -0,4,-0,2,0,0,2 et 0,4 et la structure native. Les hydrophobes pour des deltas de -0,4,-0,2,0,0,2 et 0,4 sont représentés par un jeu de couleurs allant du rouge foncé au vert clair, en passant par le jaune.

## 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, avec un premier modèle de solvant « GB », le modèle NEA et une constante diélectrique  $\epsilon_P$  égale à 8. Puis nous avons effectué de nouveaux paramétrages sur un ensemble de 3 domaines PDZ, avec une constante diélectrique  $\epsilon_P$  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 seule 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  $\epsilon_p$  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 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 proches 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 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 99 108 109 110 118 120 121 122 123 125 127 128 129 132 133 134 135 150 152
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 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	186 187 188 189 190 191 192 215 216 219 221 223 240 243 244 246 250 251 252 253 254 255 256 257 259 260 278 280 281 282
2BYG 2	1	186 187 188 189 190 191 192 215 216 219 221 223 240 243 244 245 251 252 253 254 255 256 257 259 260 278 278 280 281 282
2BYG 3	1	186 187 188 189 190 191 192 215 216 219 221 223 240 243 244 245 251 252 253 254 255 256 257 259 260 278 246 280 281 282
2BYG 4	1	186 187 188 189 190 191 192 215 216 219 221 223 240 243 244 245 246 251 252 253 254 255 256 257 259 260 278 278 281 282
2BYG 5	1	186 187 188 189 190 191 192 215 216 219 221 223 240 243 244 251 252 253 254 255 256 257 259 260 278 246 278 280 281 282

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