



Computational Protein Design: un outil pour l'ingénierie des protéines et la biologie synthétique

David Mignon

sous la direction de Thomas Simonson
Laboratoire de Biochimie, **École Polytechnique**

Computational Protein Design (CPD)

Concevoir ou modifier des protéines par informatique pour leur conférer de nouvelles propriétés

Application:

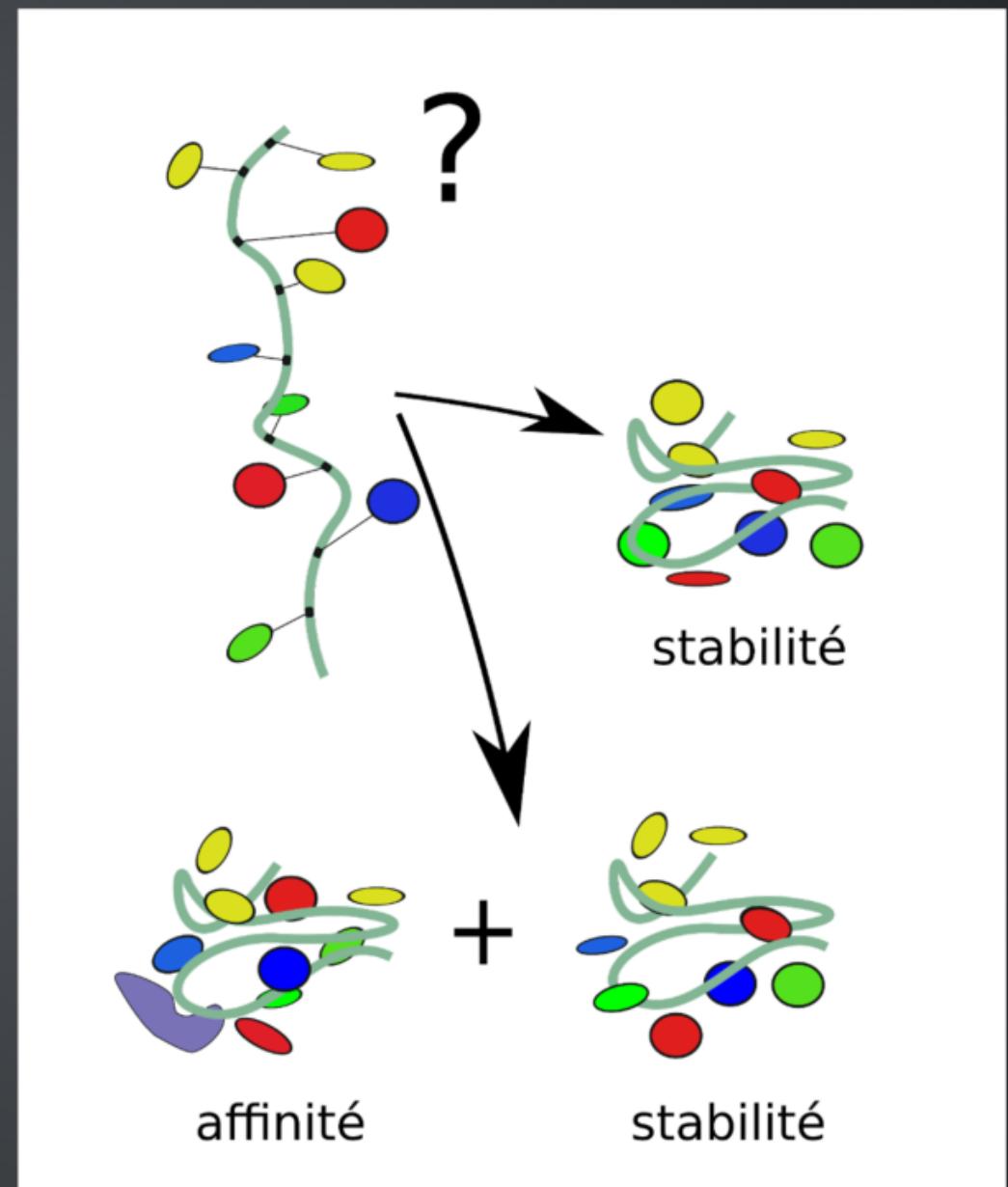
- protéines redessinées
- enzymes, complexes
- insertion d'acides aminés non-naturels

Principaux éléments:

- un espace de conformations de la protéine
- une fonction d'énergie
- un algorithme d'exploration de l'espace de séquences-conformations

Principaux programmes:

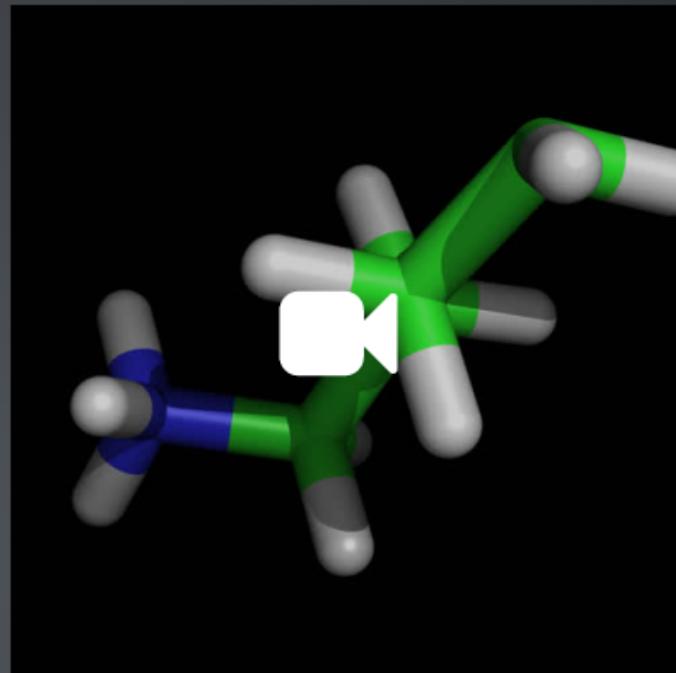
- ORBIT (Mayo,1996)
- Toulbar2 (Allouche,2014)
- Proteus (Simonson,2008)
- Rosetta (Baker,2003)



Le CPD avec Proteus

L'espace de conformation:

- Un backbone fixe
Nous utilisons le squelette d'une protéine native
- positionnent des chaînes latérales discrétisées (rotamères)
Utilisation d'un bibliothèque de rotamères:Tuffery95
- Les prolines et les glycines natives sont conservées



L'état déplié: l'énergie de référence
Pour une séquence S de type t_i

$$E^u(S) = \sum_i^N E_{t_i}^u$$

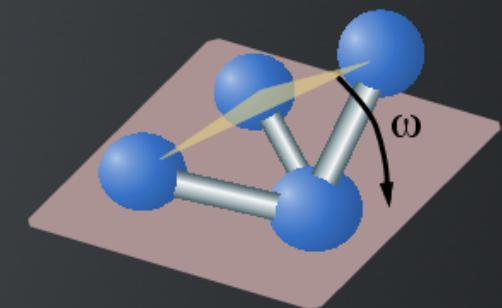
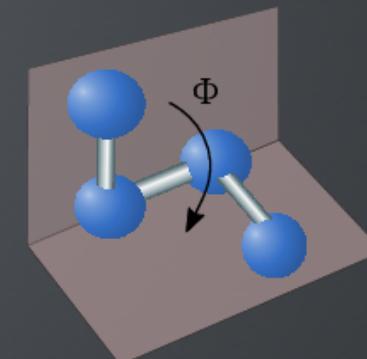
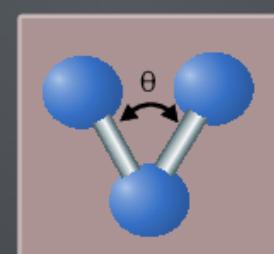
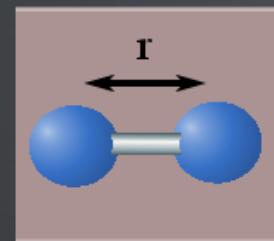
La fonction d'énergie

L'énergie interne à la protéine:

Utilisation de la mécanique moléculaires avec le champ de force Amber

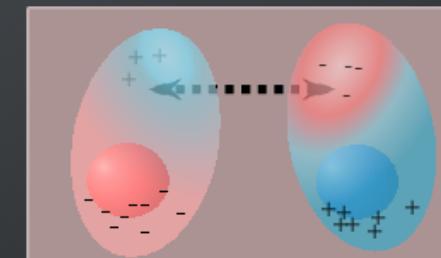
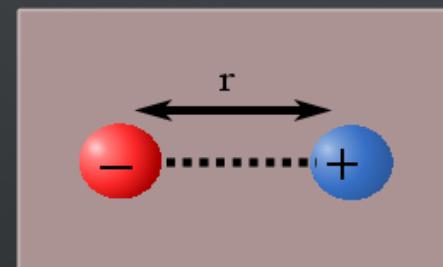
- Les interactions liées:

$$E_{liée} = E_{liaison} + E_{angle} + E_{diedre} + E_{impr}$$



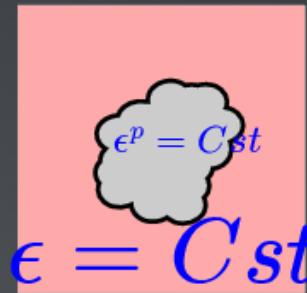
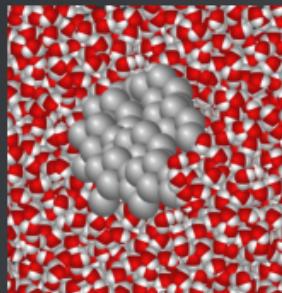
- Les interactions non liées

$$E_{non \ liées} = E_{elec} + E_{vdw}$$



La fonction d'énergie

Les modèles de solvant implicites



1. l'effet hydrophobe:

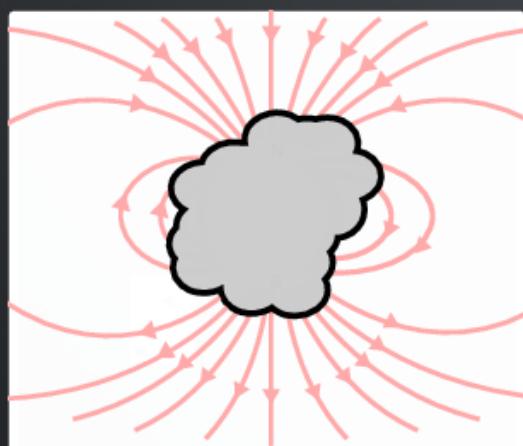
modèle Surface Area (SA)

$$E_{hydro} \approx \sum_i \sigma_{t_i} A_i$$

2. Traitement des interactions électrostatiques:

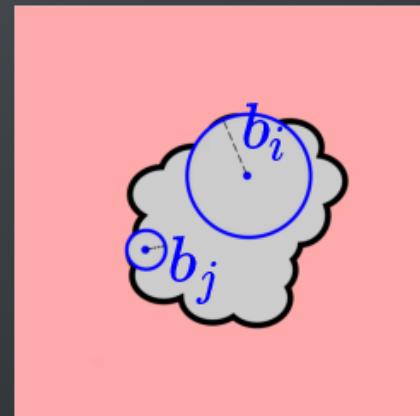
Coulomb

$$E_{solv}^{elec} \approx \left(\frac{1}{\epsilon} - 1\right) E_{coul}^p$$



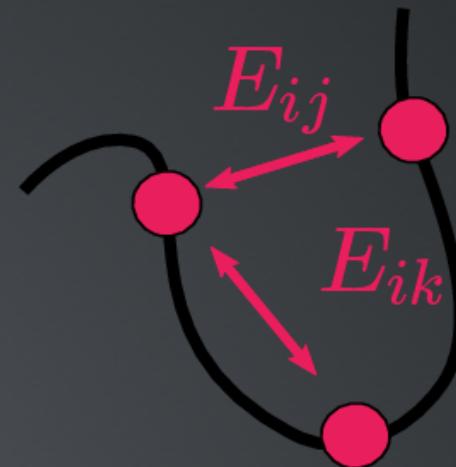
Generalised Born (GB)

$$E_{tot}^{elec} = \frac{1}{2} \sum_{i \neq j} \frac{q_i q_j}{\epsilon_s r_{ij}} - \frac{1}{2} \tau \sum_{i,j}^N g_{ij}(b_i, b_j)$$



Décomposition par paires de la fonction d'énergie

$$E(C) = \sum_i E_i + \sum_{i \neq j} E_{ij}$$



- rendre possible l'utilisation de plusieurs algorithmes
- accélérer de l'étape d'exploration par pré-calculation des interactions -> stockage dans une matrice.
- nécessite des approximations supplémentaires:
 1. Pour la décomposition du terme surfacique
 2. Pour la décomposition de l'environnement GB

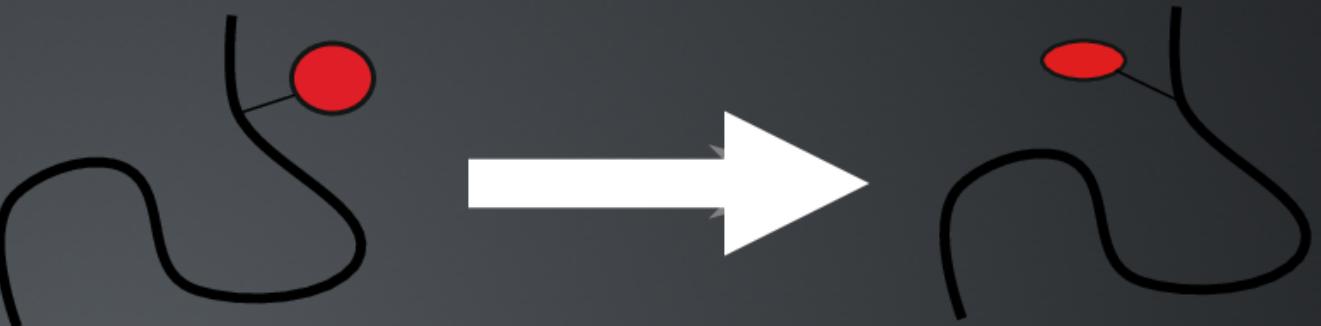
Première méthode: Native Environnement Approximation (NEA)

Les rayons de solvatation d'une chaîne latérale sont calculés en fixant tout le reste du système dans sa séquence et sa conformation native.

Principe de l'exploration

Déplacement dans l'espace des conformations:

modification du rotamère à une position i sur la protéine repliée

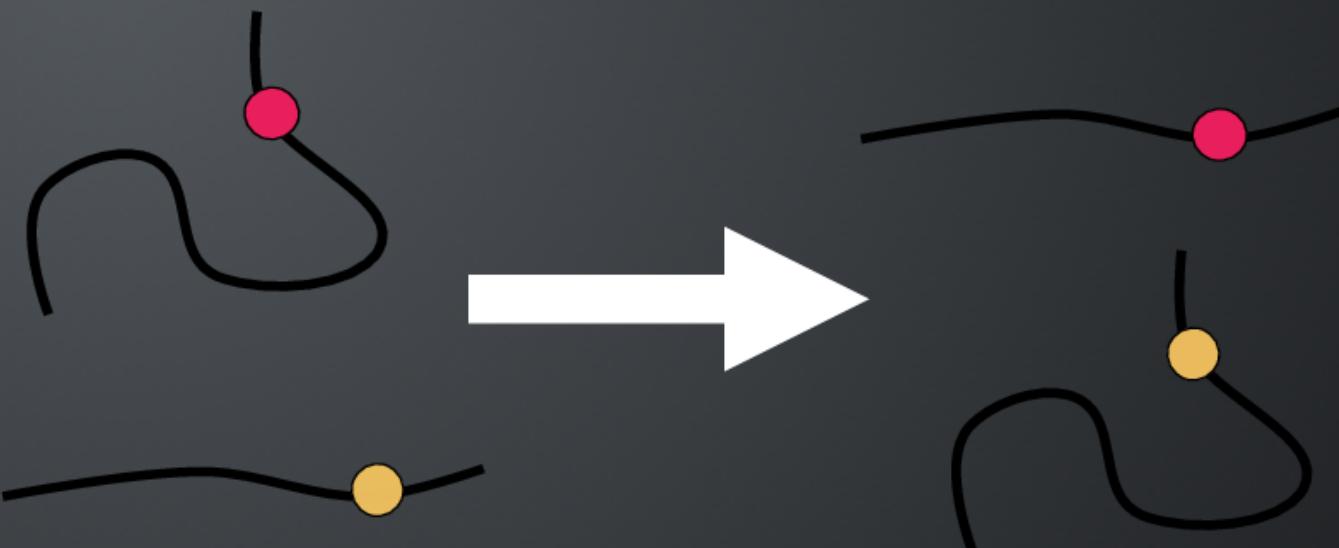


$$\Delta E = E(.., rot_i^{new}, ..) - E(.., rot_i^{old}, ..)$$

Déplacement dans l'espace des séquences:

modification du type de chaîne latérale à une position i sur la protéine repliée.

En même temps, une mutation inverse sur la protéine dépliée, en i



$$\Delta E = \Delta E_f - \Delta E_{uf}$$

Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

Pour chaque cycle heuristique
une séquence-conformation **S** est choisie aléatoirement
Tant que l'énergie de **S** est améliorée
Pour **i** allant de la première position de **S** jusqu'à la dernière
S est fixée sauf à la position **i**
le meilleur rotamère possible en **i** est déterminé
Ce rotamère est utilisé pour fixer **S** en **i**
Fin de Pour
Fin de Tant que
S est sauvegardée
Fin du cycle heuristique

Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

Pour chaque cycle heuristique

une séquence-conformation **S** est choisie aléatoirement

Tant que l'énergie de **S** est améliorée

Pour **i** allant de la première position de **S** jusqu'à la dernière

S est fixée sauf à la position **i**

le meilleur rotamère possible en **i** est déterminé

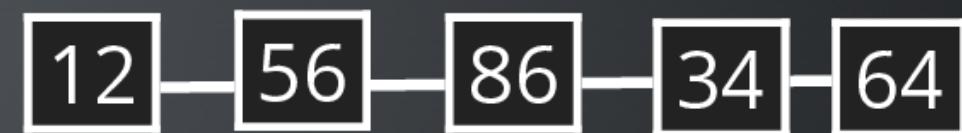
Ce rotamère est utilisé pour fixer **S** en **i**

Fin de Pour

Fin de Tant que

S est sauvegardée

Fin du cycle heuristique



Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

Pour chaque cycle heuristique
une séquence-conformation **S** est choisie aléatoirement

Tant que l'énergie de **S** est améliorée

Pour **i** allant de la première position de **S** jusqu'à la dernière

S est fixée sauf à la position **i**

le meilleur rotamère possible en **i** est déterminé

Ce rotamère est utilisé pour fixer **S** en **i**

Fin de Pour

Fin de Tant que

S est sauvegardée

Fin du cycle heuristique



Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

Pour chaque cycle heuristique
une séquence-conformation **S** est choisie aléatoirement
Tant que l'énergie de **S** est améliorée
Pour i allant de la première position de S jusqu'à la dernière
S est fixée sauf à la position **i**
le meilleur rotamère possible en **i** est déterminé
Ce rotamère est utilisé pour fixer **S** en **i**
Fin de Pour
Fin de Tant que
S est sauvegardée
Fin du cycle heuristique



Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

Pour chaque cycle heuristique
une séquence-conformation **S** est choisie aléatoirement
Tant que l'énergie de **S** est améliorée
Pour **i** allant de la première position de **S** jusqu'à la dernière
S est fixée sauf à la position **i**
le meilleur rotamère possible en **i** est déterminé
Ce rotamère est utilisé pour fixer **S** en **i**
Fin de Pour
Fin de Tant que
S est sauvegardée
Fin du cycle heuristique



Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

Pour chaque cycle heuristique

une séquence-conformation **S** est choisie aléatoirement

Tant que l'énergie de **S** est améliorée

Pour **i** allant de la première position de **S** jusqu'à la dernière

S est fixée sauf à la position **i**

le meilleur rotamère possible en **i** est déterminé

Ce rotamère est utilisé pour fixer **S** en **i**

Fin de Pour

Fin de Tant que

S est sauvegardée

Fin du cycle heuristique



Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

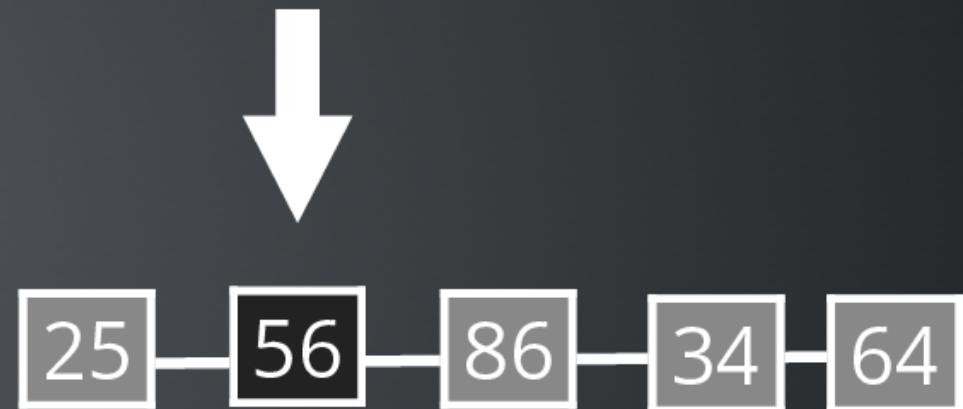
Pour chaque cycle heuristique
une séquence-conformation **S** est choisie aléatoirement
Tant que l'énergie de **S** est améliorée
Pour **i** allant de la première position de **S** jusqu'à la dernière
S est fixée sauf à la position **i**
le meilleur rotamère possible en **i** est déterminé.
Ce rotamère est utilisé pour fixer **S en **i****
Fin de Pour
Fin de Tant que
S est sauvegardée
Fin du cycle heuristique



Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

Pour chaque cycle heuristique
une séquence-conformation **S** est choisie aléatoirement
Tant que l'énergie de **S** est améliorée
Pour i allant de la première position de S jusqu'à la dernière
S est fixée sauf à la position **i**
le meilleur rotamère possible en **i** est déterminé.
Ce rotamère est utilisé pour fixer **S** en **i**
Fin de Pour
Fin de Tant que
S est sauvegardée
Fin du cycle heuristique



Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

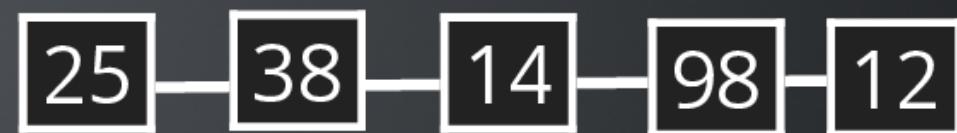
Pour chaque cycle heuristique
une séquence-conformation **S** est choisie aléatoirement
Tant que l'énergie de **S** est améliorée
Pour **i** allant de la première position de **S** jusqu'à la dernière
S est fixée sauf à la position **i**
le meilleur rotamère possible en **i** est déterminé
Ce rotamère est utilisé pour fixer **S** en **i**
Fin de Pour
Fin de Tant que
S est sauvegardée
Fin du cycle heuristique



Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

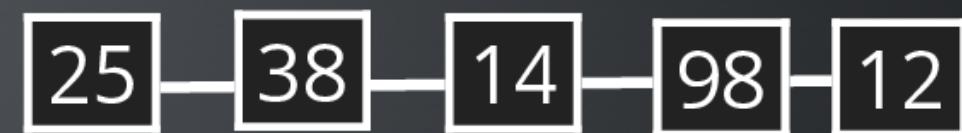
Pour chaque cycle heuristique
une séquence-conformation **S** est choisie aléatoirement
Tant que l'énergie de **S** est améliorée
Pour **i** allant de la première position de **S** jusqu'à la dernière
S est fixée sauf à la position **i**
le meilleur rotamère possible en **i** est déterminé
Ce rotamère est utilisé pour fixer **S** en **i**
Fin de Pour
Fin de Tant que
S est sauvegardée
Fin du cycle heuristique



Le "Multistart Steepest Descent" (Wernisch,Wodak)

Une heuristique spécifique à l'espace d'état

Pour chaque cycle heuristique
une séquence-conformation **S** est choisie aléatoirement
Tant que l'énergie de **S** est améliorée
Pour **i** allant de la première position de **S** jusqu'à la dernière
S est fixée sauf à la position **i**
le meilleur rotamère possible en **i** est déterminé
Ce rotamère est utilisé pour fixer **S** en **i**
Fin de Pour
Fin de Tant que
S est sauvegardée
Fin du cycle heuristique

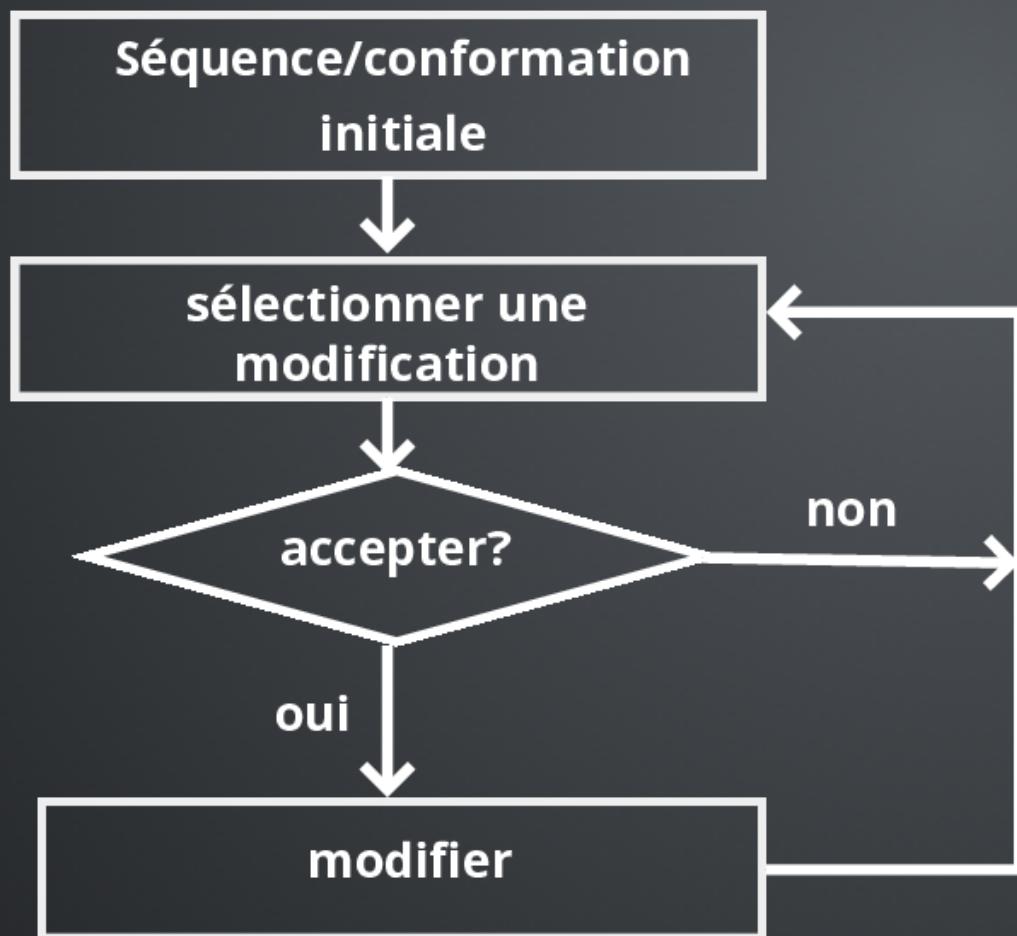


Le Monte Carlo

algorithme Metropolis-Hastings

L'objectif est de générer une collection d'états échantillonés selon la distribution de Boltzmann:

$$\mathcal{P}(x) = \frac{1}{Z} e^{-\frac{E}{RT}}$$



Un pas depuis l'état x est contrôlé par 2 probabilités conditionnelles:

- $\sigma(\cdot|x)$ pour le choix de la modification
- $\alpha(\cdot|x)$ pour l'accepter

Le Monte Carlo

algorithme Metropolis-Hastings

L'algorithme définit une chaîne de Markov

Si la balance détaillée est respectée

$$\mathcal{P}(x \rightarrow y) = \mathcal{P}(y \rightarrow x)$$

Alors la chaîne possède une distribution stationnaire.

Si la chaîne est ergodique, elle converge vers cette distribution.

Metropolis propose pour σ symétrique:

$$\alpha(y|x) = \min\left\{1, \frac{\mathcal{P}(y)}{\mathcal{P}(x)}\right\} = \min\left\{1, e^{(-\frac{\Delta E}{RT})}\right\}$$

Le Monte-Carlo

Algorithme Metropolis-Hastings

La distribution cible étant donnée $p(\cdot)$
Une séquence-conformation S est choisie aléatoirement
Pour chaque pas de la trajectoire

Une modification possible est sélectionnée à partir d'une distribution conditionnelle:

$$S' \sim \sigma(S'|S)$$

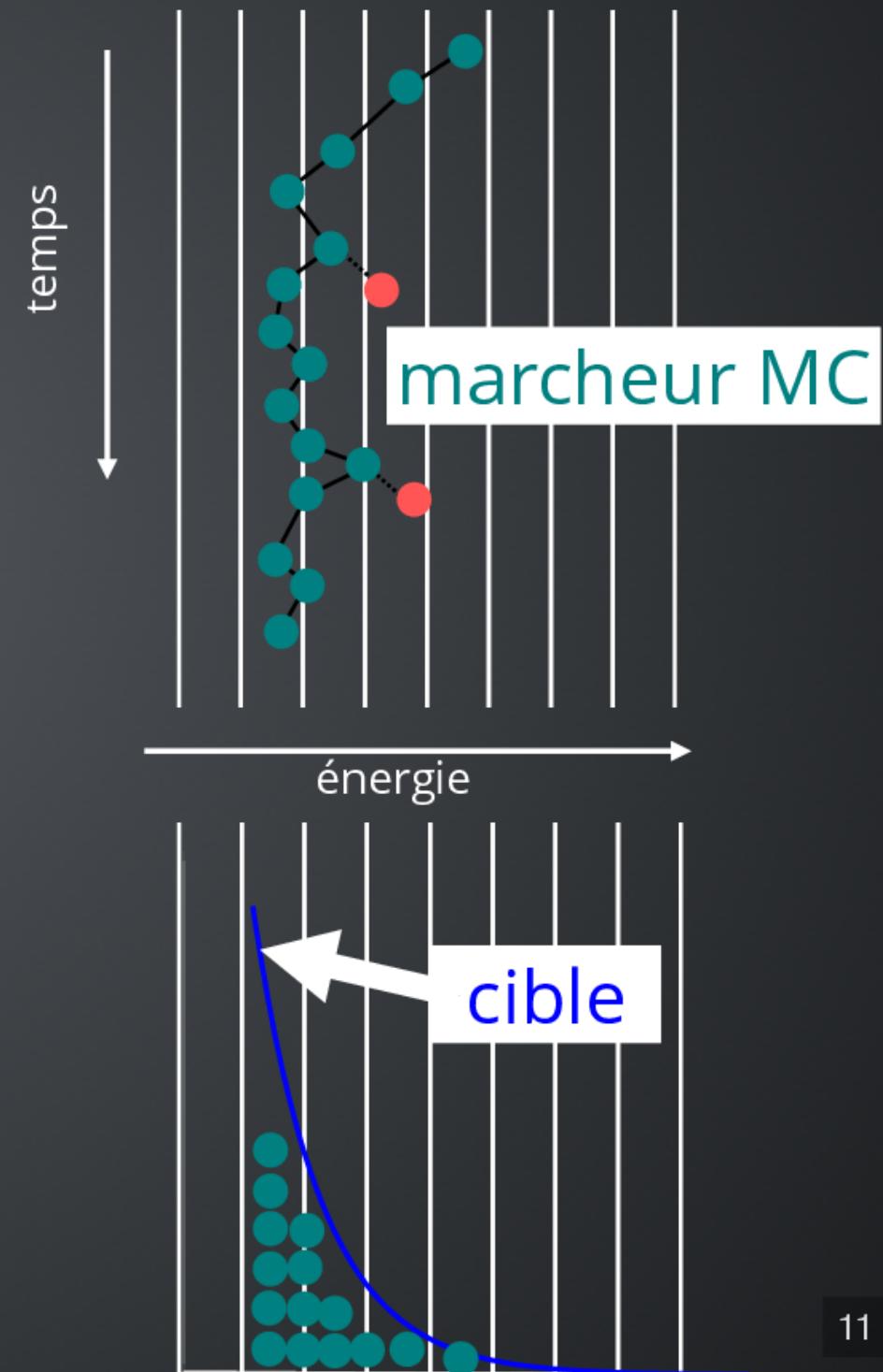
La modification est acceptée selon la probabilité:

$$\alpha(S'|S) = \min\{1, e^{(-\frac{\Delta E}{RT})} \frac{\sigma(S|S')}{\sigma(S'|S)}\}$$

Fin de Pour

indépendant de la fonction de partition

généralisation d'Hastings



Replica Exchange Monte Carlo

accélérer la convergence en visitant plusieurs zones énergétiques simultanément

Lancement en parallèle de N marcheurs Monte Carlo aux températures ordonnées (t_1, \dots, t_n)

Périodiquement un couple de marcheurs aux températures (t_i, t_{i+1}) est sélectionnés aléatoirement.

Les températures entre les deux marcheurs sont échangées selon la probabilité:

$$\beta = \min\left\{1, \exp\left(-\left(\frac{1}{kt_i} - \frac{1}{kt_{i+1}}\right)(E_{t_i} - E_{t_{i+1}})\right)\right\}$$

- L'échange de températures est une mouvement supplémentaire dans l'espace généralisé des N répliques
- β est l'expression de la balance détaillée pour ce mouvement

Donc les propriétés Monte Carlo de la trajectoire sont conservées.

Une méthode exacte par optimisation combinatoire (Toulbar2)

L'algorithme "Depth-First Branch and Bound"

Le principe de séparation: partitionner l'ensemble des séquences-conformations en sous-ensemble fils

→ construction d'un arbre

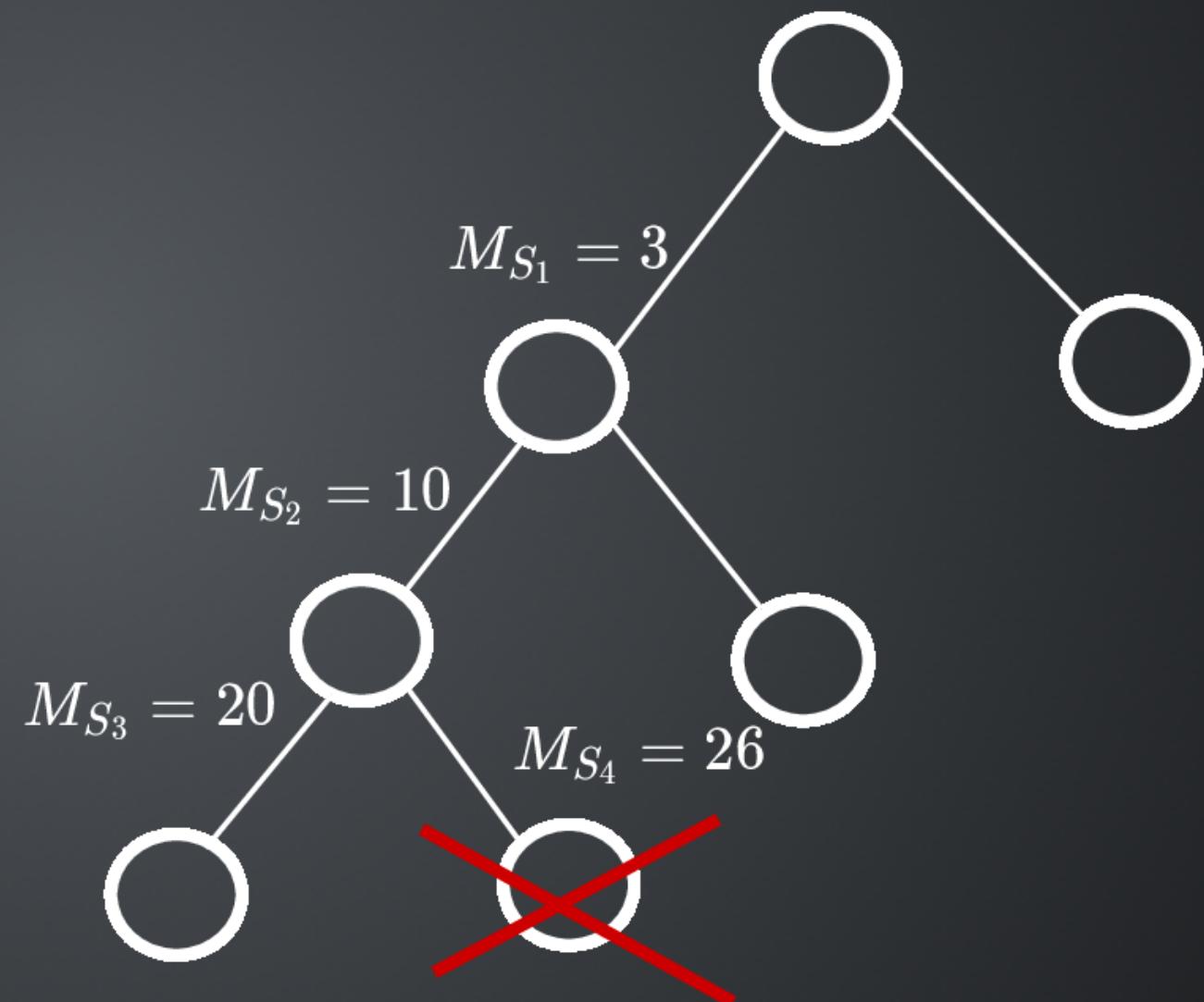
Le développement de l'arbre: descendre dans les branches autant que possible, sinon remonter d'un sommet.

L'EPT produit un **minorant** M_S des coûts d'un sommet.

Si le meilleur coût connu est inférieur à un minorant d'un sommet S , on peut élaguer l'arbre en S .

Mise à jour des minorants

$$M_{S_0} = 0$$



Une méthode exacte par optimisation combinatoire (Toulbar2)

Equivalence Preserving Transformation (EPT)
pour accélérer le DFBB

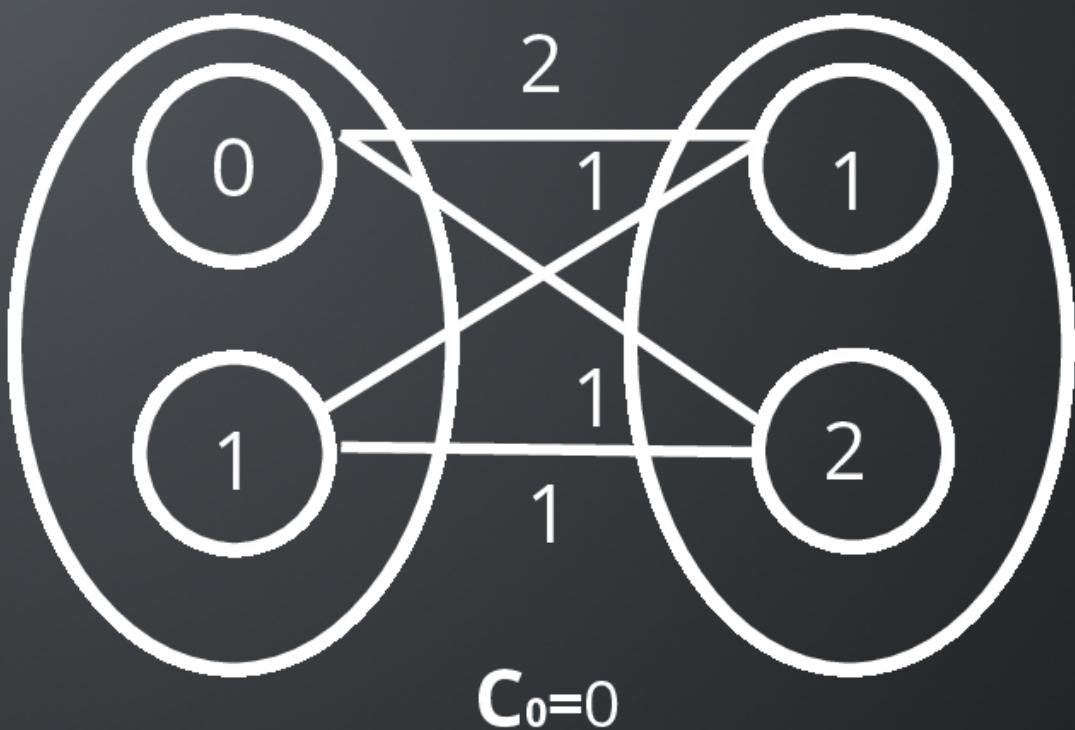
La décomposition par paire de notre fonction d'énergie permet une représentation sous forme d'un réseau de fonctions de coûts

- Une interaction entre acides aminés \Leftrightarrow une arête du réseau
- Une énergie d'un rotamère \Leftrightarrow un nœud du réseau

Deux transformations de base:

- la projection
- la distribution

"Dead-End Elimination" en complément



Une méthode exacte par optimisation combinatoire (Toulbar2)

Equivalence Preserving Transformation (EPT)
pour accélérer le DFBB

La décomposition par paire de notre fonction d'énergie permet une représentation sous forme d'un réseau de fonctions de coûts

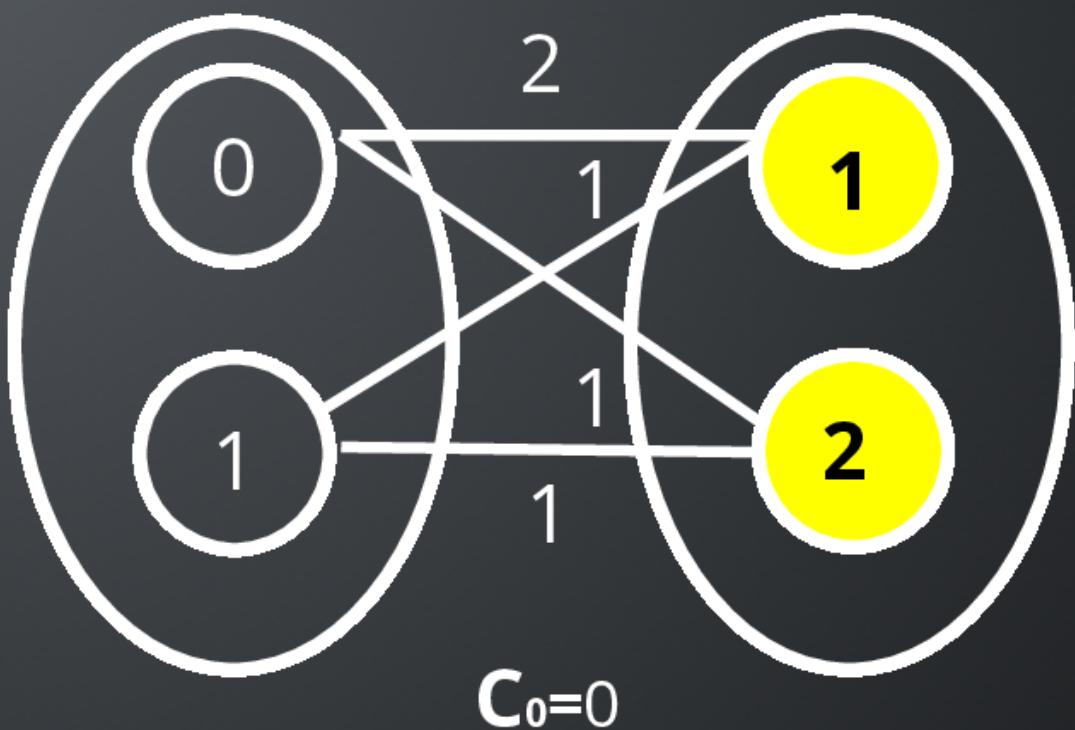
- Une interaction entre acides aminés \Leftrightarrow une arête du réseau
- Une énergie d'un rotamère \Leftrightarrow un nœud du réseau

Deux transformations de base:

- **la projection**

- la distribution

"Dead-End Elimination" en complément



Une méthode exacte par optimisation combinatoire (Toulbar2)

Equivalence Preserving Transformation (EPT)
pour accélérer le DFBB

La décomposition par paire de notre fonction d'énergie permet une représentation sous forme d'un réseau de fonctions de coûts

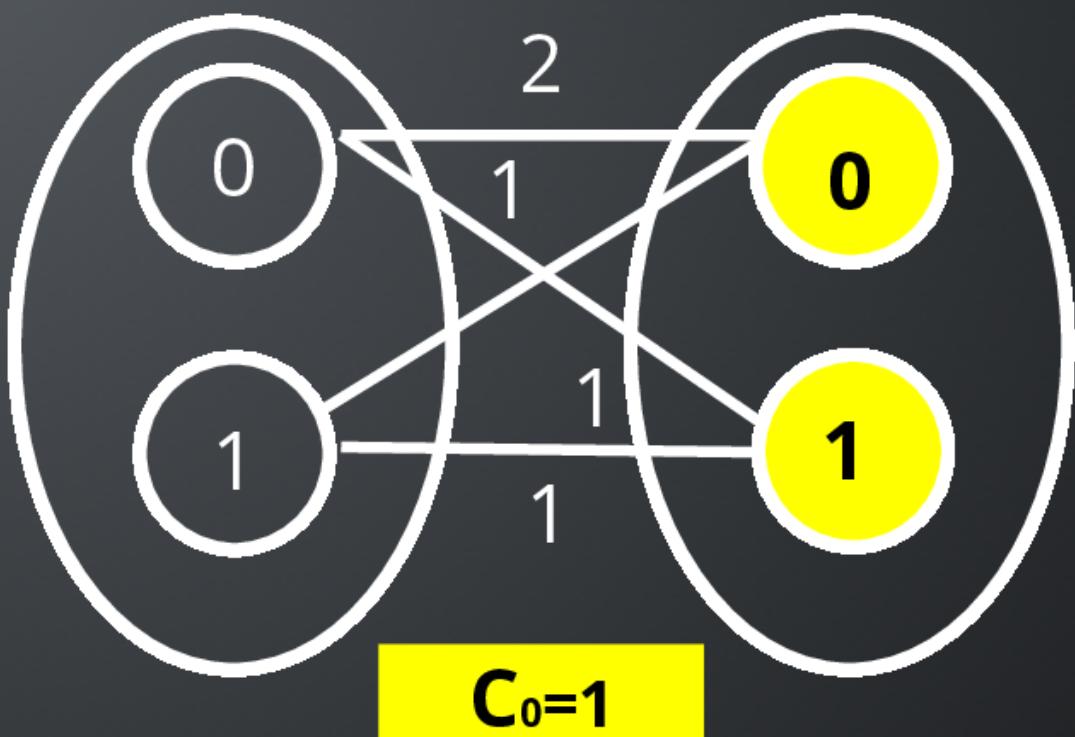
- Une interaction entre acides aminés \Leftrightarrow une arête du réseau
- Une énergie d'un rotamère \Leftrightarrow un nœud du réseau

Deux transformations de base:

- **la projection**

- la distribution

"Dead-End Elimination" en complément



Une méthode exacte par optimisation combinatoire (Toulbar2)

Equivalence Preserving Transformation (EPT)
pour accélérer le DFBB

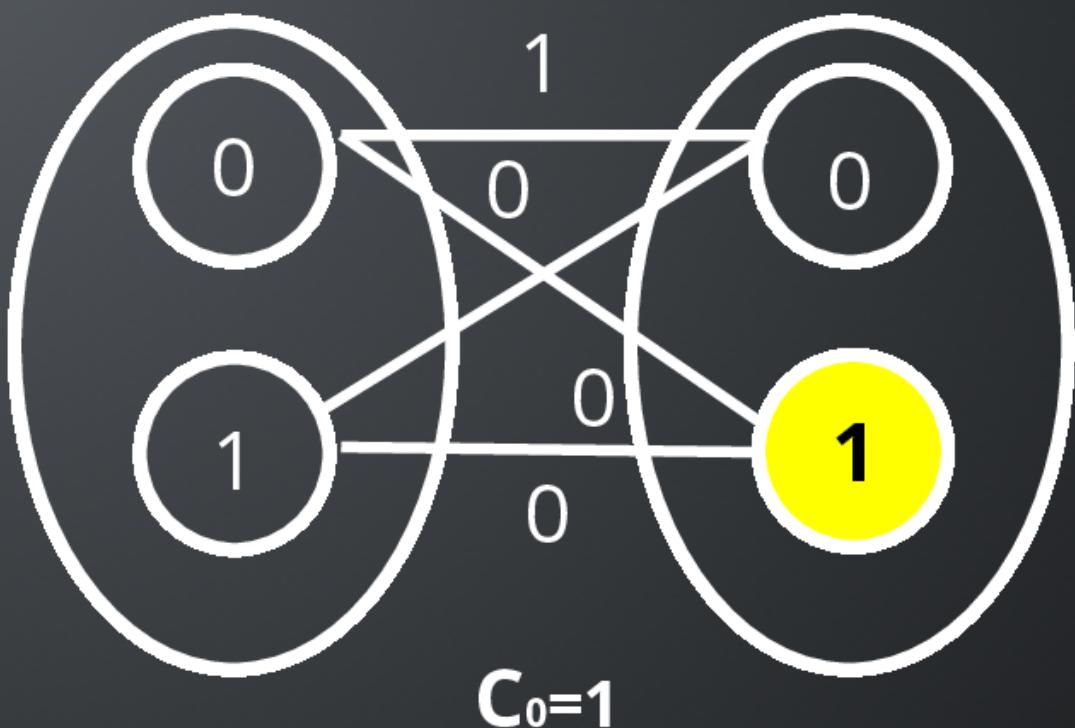
La décomposition par paire de notre fonction d'énergie permet une représentation sous forme d'un réseau de fonctions de coûts

- Une interaction entre acides aminés \Leftrightarrow une arête du réseau
- Une énergie d'un rotamère \Leftrightarrow un nœud du réseau

Deux transformations de base:

- la projection
- **la distribution**

"Dead-End Elimination" en complément



Une méthode exacte par optimisation combinatoire (Toulbar2)

Equivalence Preserving Transformation (EPT)
pour accélérer le DFBB

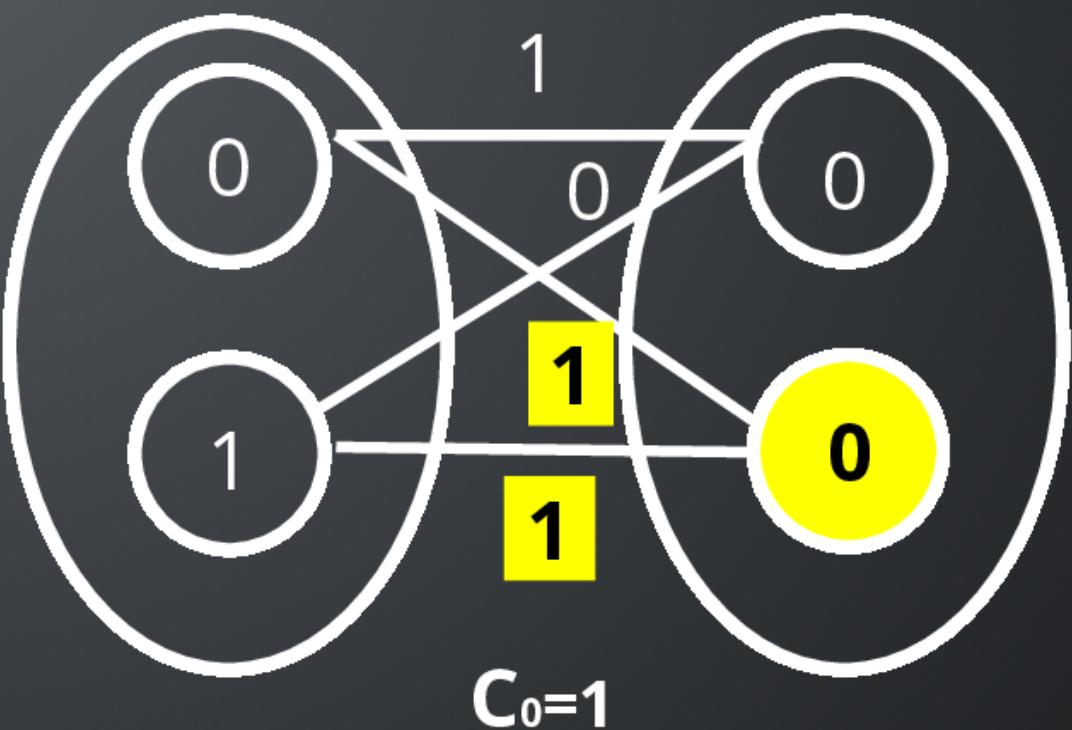
La décomposition par paire de notre fonction d'énergie permet une représentation sous forme d'un réseau de fonctions de coûts

- Une interaction entre acides aminés \Leftrightarrow une arête du réseau
- Une énergie d'un rotamère \Leftrightarrow un nœud du réseau

Deux transformations de base:

- la projection
- **la distribution**

"Dead-End Elimination" en complément



Une méthode exacte par optimisation combinatoire (Toulbar2)

Equivalence Preserving Transformation (EPT)
pour accélérer le DFBB

La décomposition par paire de notre fonction d'énergie permet une représentation sous forme d'un réseau de fonctions de coûts

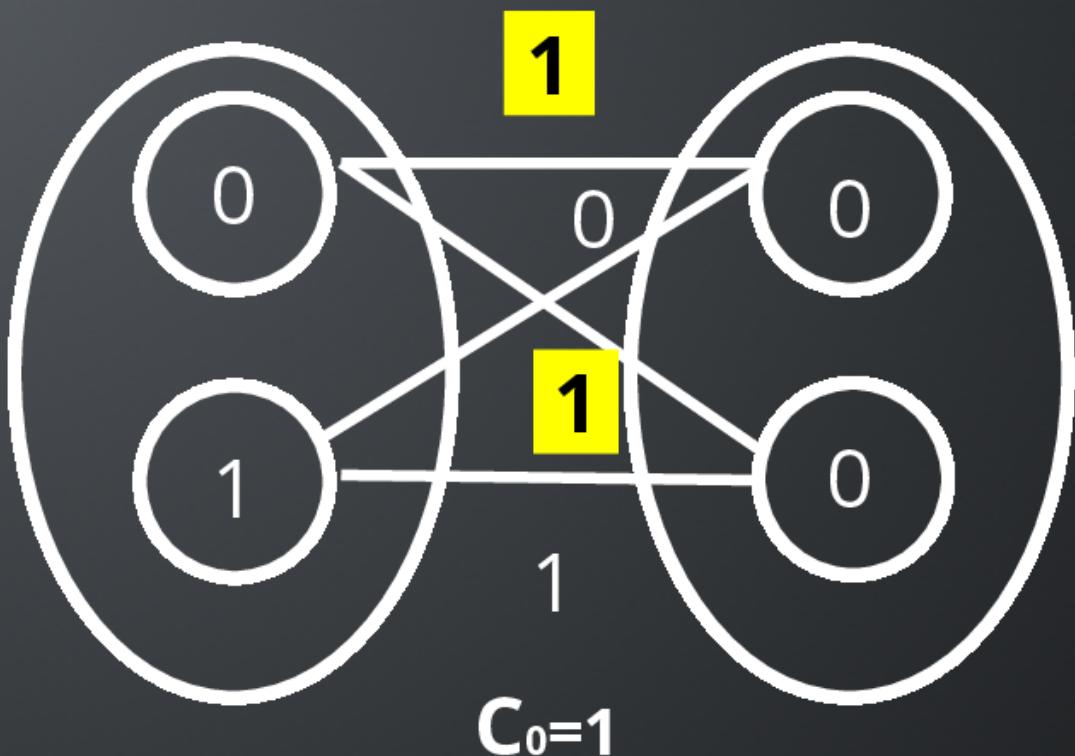
- Une interaction entre acides aminés \Leftrightarrow une arête du réseau
- Une énergie d'un rotamère \Leftrightarrow un nœud du réseau

Deux transformations de base:

- **la projection**

- la distribution

"Dead-End Elimination" en complément



Une méthode exacte par optimisation combinatoire (Toulbar2)

Equivalence Preserving Transformation (EPT)
pour accélérer le DFBB

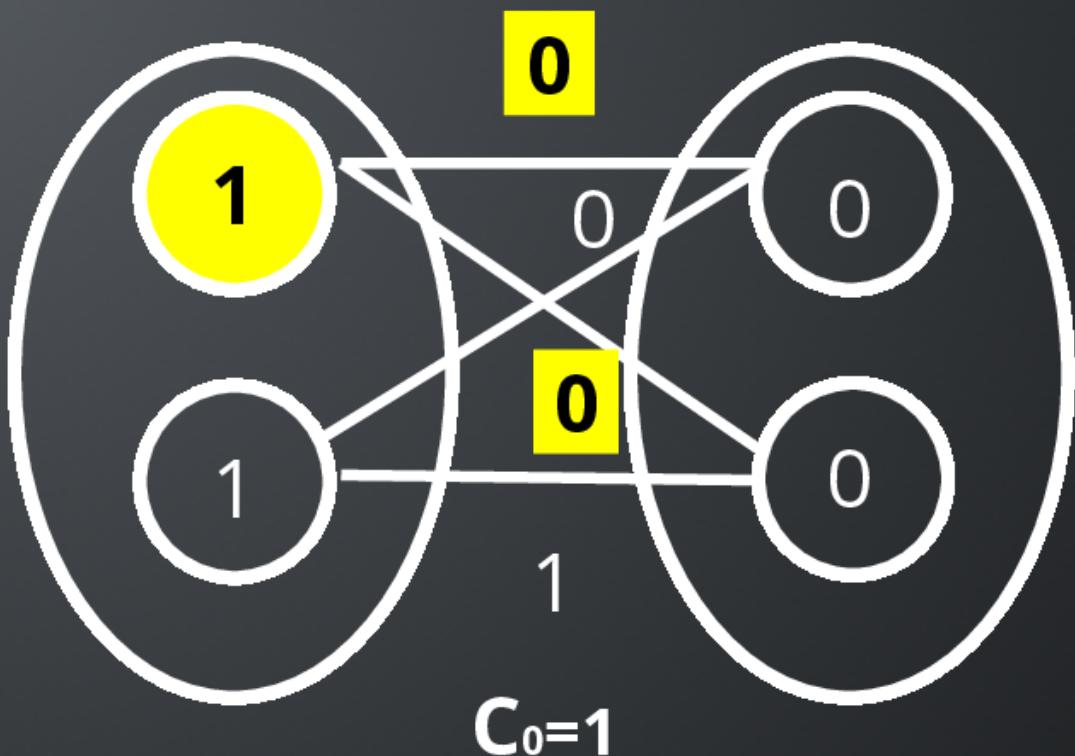
La décomposition par paire de notre fonction d'énergie permet une représentation sous forme d'un réseau de fonctions de coûts

- Une interaction entre acides aminés \Leftrightarrow une arête du réseau
- Une énergie d'un rotamère \Leftrightarrow un nœud du réseau

Deux transformations de base:

- **la projection**
- la distribution

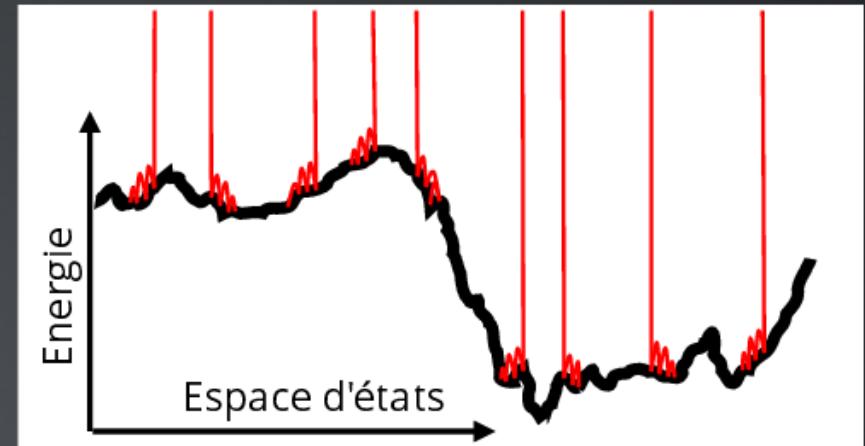
"Dead-End Elimination" en complément



Différentes approches

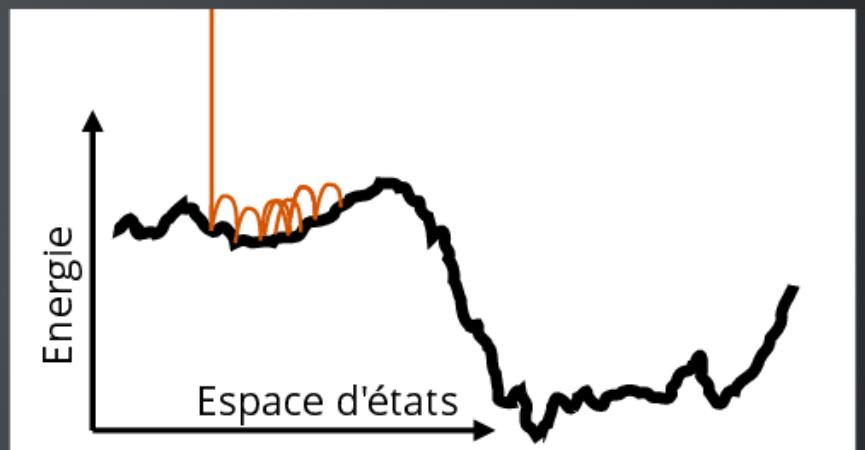
MSD

- Pour un cycle, l'exploration est limitée.
- Mais un cycle est rapide.



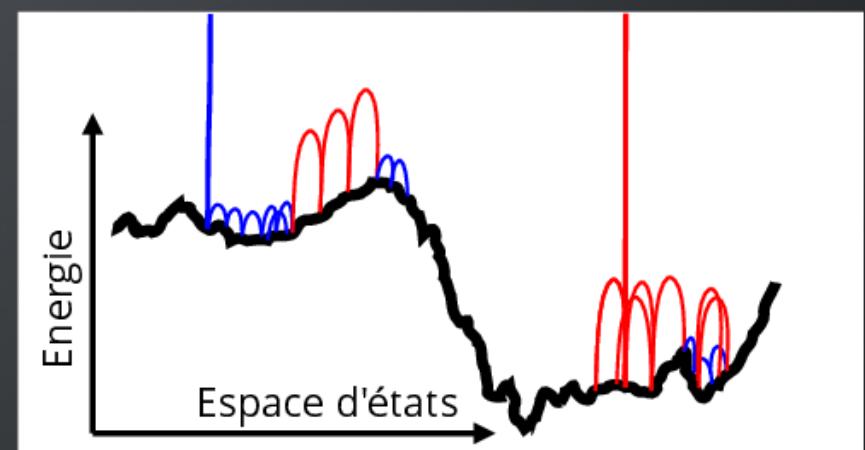
Monte Carlo

- Le marcheur peut tout visiter.
- Mais la convergence est lente.



REMC

- Il conserve les propriétés physique de la trajectoire.
- La convergence est accélérée.



Comparaison D' algorithmes d'exploration

Comparaison des algorithmes

L'ensemble de tests

Systèmes

Modèle

Protéine	nb résidus	famille	Modèle
1A81	108	SH2	• Amber (ff99SB)
1BM2	98	SH2	• CASA, $\epsilon = 23$
1M61	109	SH2	• toutes les positions mutables, sauf GLY et PRO
1O4C	104	SH2	
1ABO	58	SH3	• énergies de références optimisées sur les 3 familles
1CKA	57	SH3	
1G9O	91	PDZ	
1R6J	82	PDZ	
2BYG	97	PDZ	

Comparaison des algorithmes

méthodes

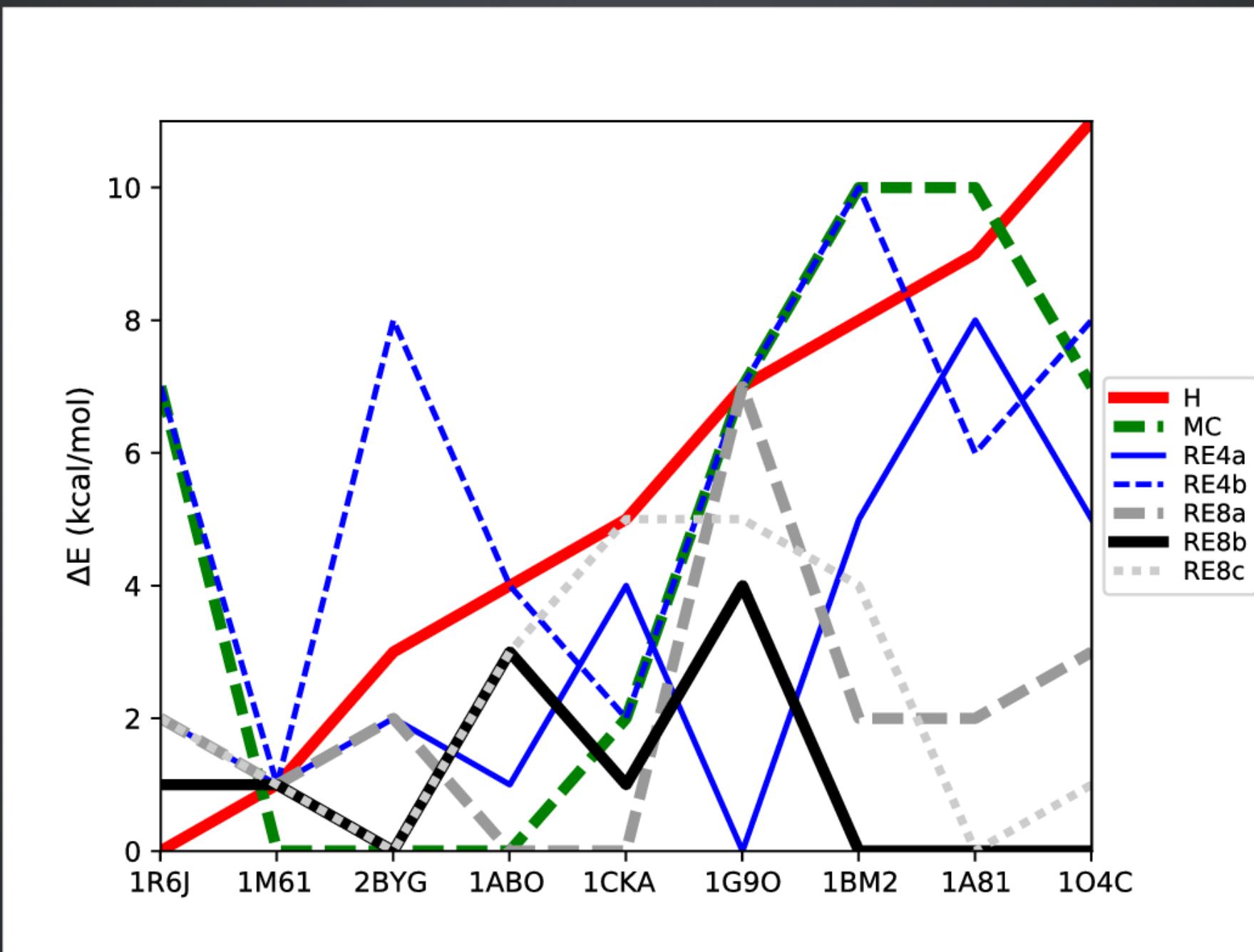
temps d'exécution limité à 24h

- Heuristique: 110 000 cycles
- Monte carlo: 6 milliards de pas
- REMC: 6 milliards de pas cumulés sur les marcheurs

Paramétrages

Algo	nb marcheurs	températur es	mutation / pas	change de rotamères/ pas	freq swap
MC	1	0,2	1,1	0,1	-
REMC	4	0,125...1	0,1	1,1	0,005
REMC	4	0,25...2	0,1	1,1	0,005
REMC	8	0,175...3	1,1	0,1	0,01
REMC	8	0,175...3	0,1	1,1	0,01
REMC	8	0,175...3	0,1	1,1	0,001

Comparaison des meilleures énergies de chaque protocole

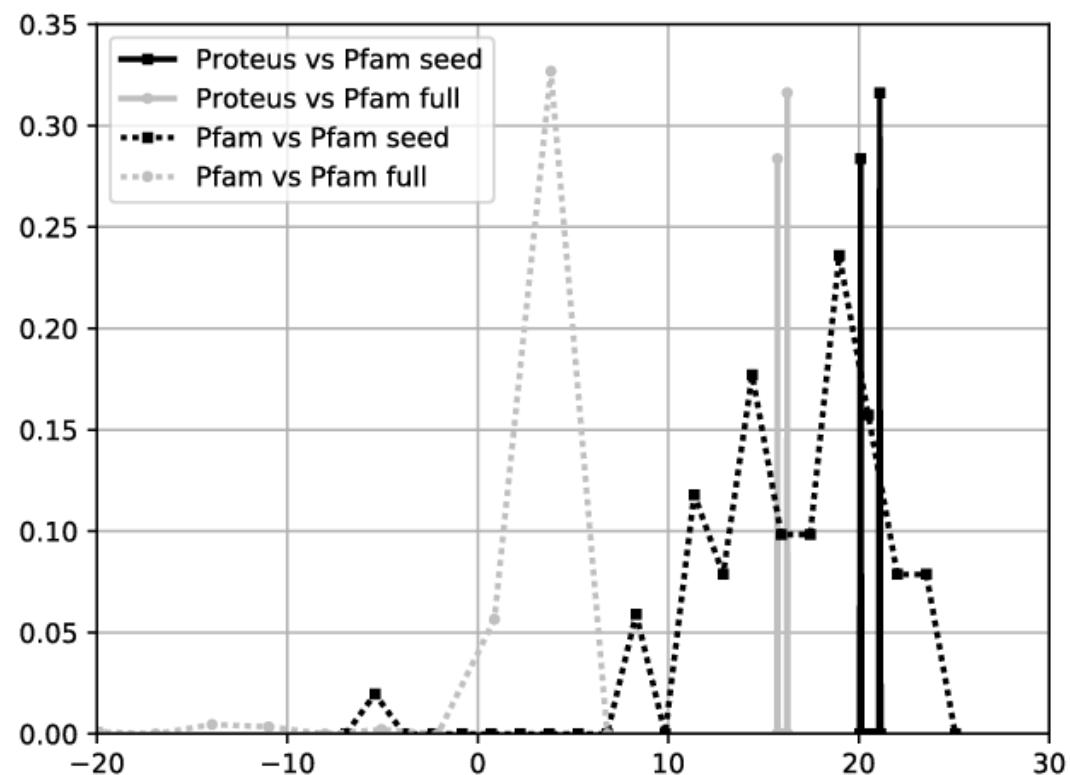


Caractérisation des séquences

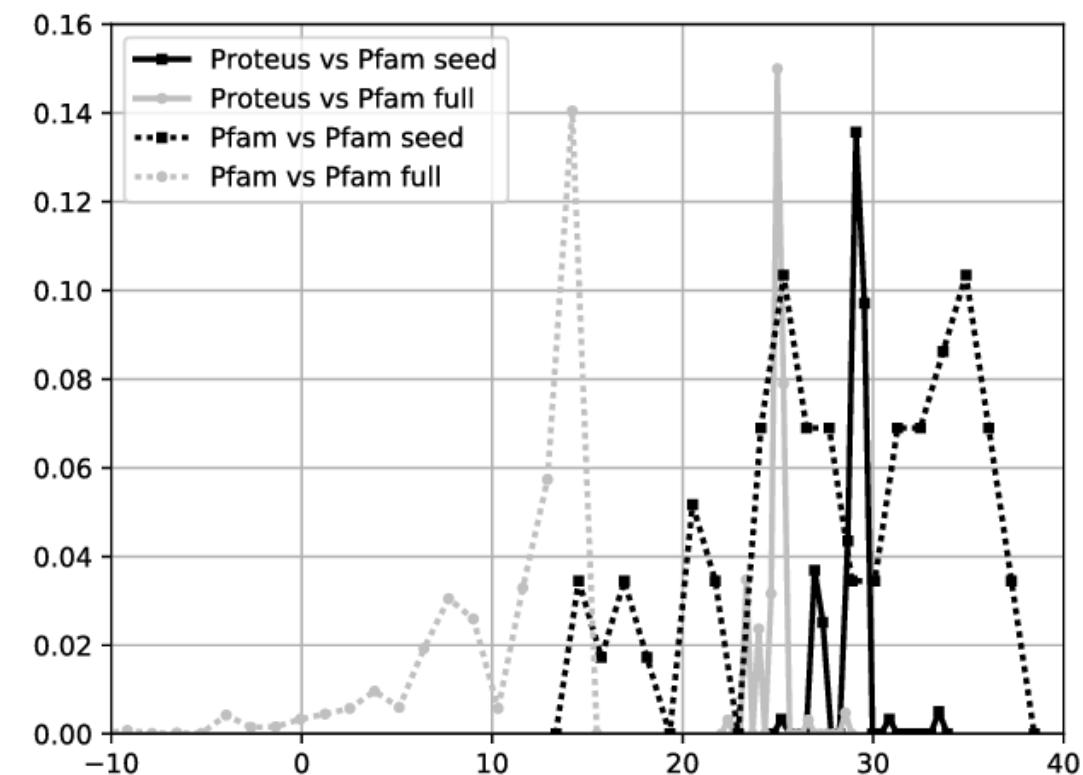
- calcul de similarité aux alignements "seed" et "full" de Pfam (Protein families database) de la famille correspondante, grâce à la matrice BLOSUM64, aux positions du cœur.
- soumission à Superfamily reconnaissance de structure 3D à partir d'une bibliothèque de HMM obtenue à partir de la classification SCOP
- Taux d'identité à la séquence native (backbone)
- Entropie résiduel par position comme mesure de la diversité

Scores de similarité sur les positions du cœur

1ABO



1BM2



Résultats Superfamily et identité

sur les 10000 séquences-rotamères de meilleures énergies

Protéine	nb de séquences	% identité à la native	taille du "match"	E-value Super-famille	% succès Super-famille	E-value famille	taux succès famille
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%	4.6e-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%

Recherche du GMEC

l'Espace est réduit progressivement en fixant une partie des résidus avec leur type natif:

- Tests à 30, 20, 10 et 5 positions actives
- Dans chaque cas 5 sélections en privilégiant les ensembles en interaction pour chacune des 9 protéines.

CFN: Temps d'exécution max 24h, en cas d'échec relance avec une seconde configuration

Nos algorithmes: MSD et le meilleur RMEC

Recherche du GMEC

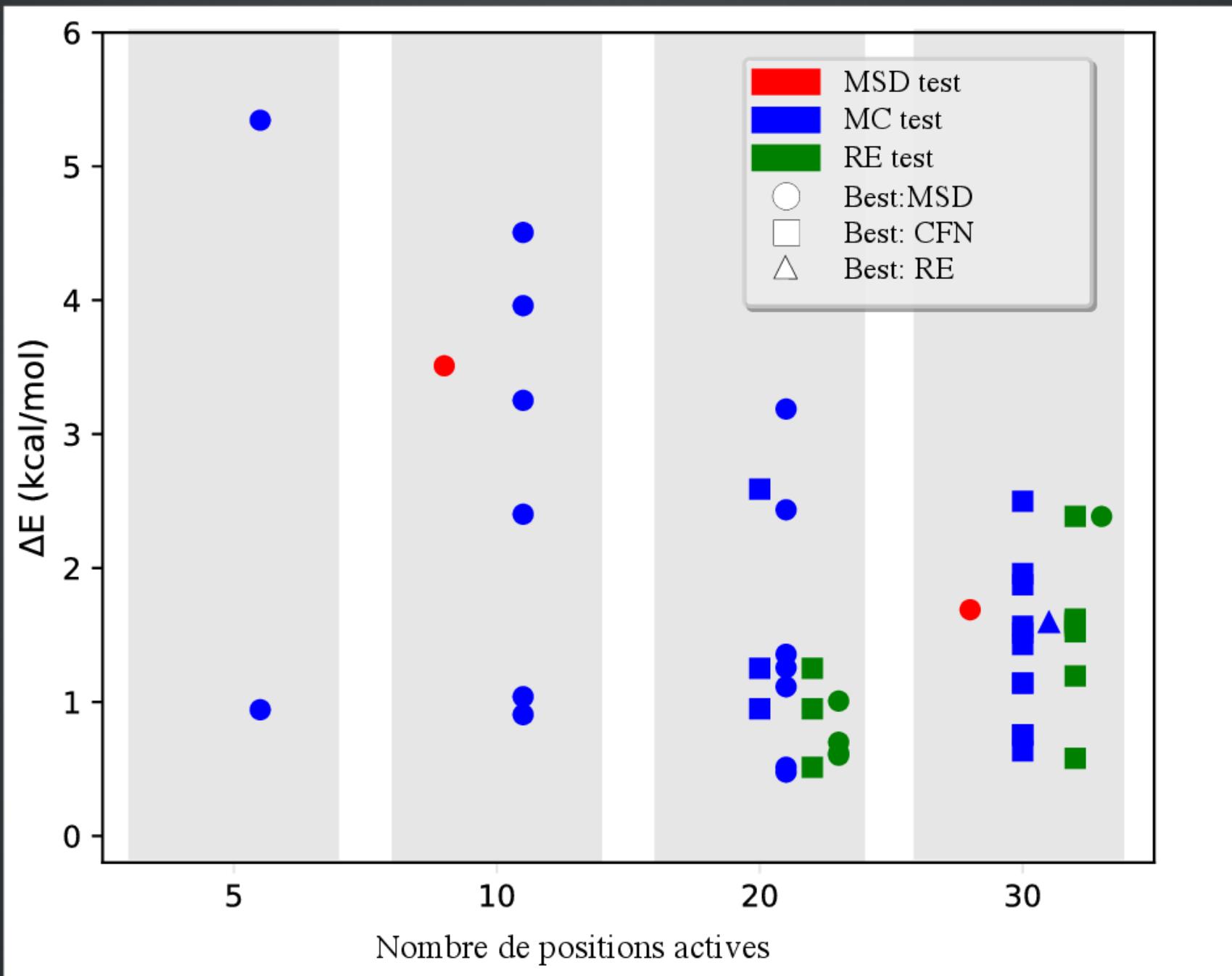
Résultats

l'Espace est réduit progressivement en fixant une partie des résidus avec leur type natif:

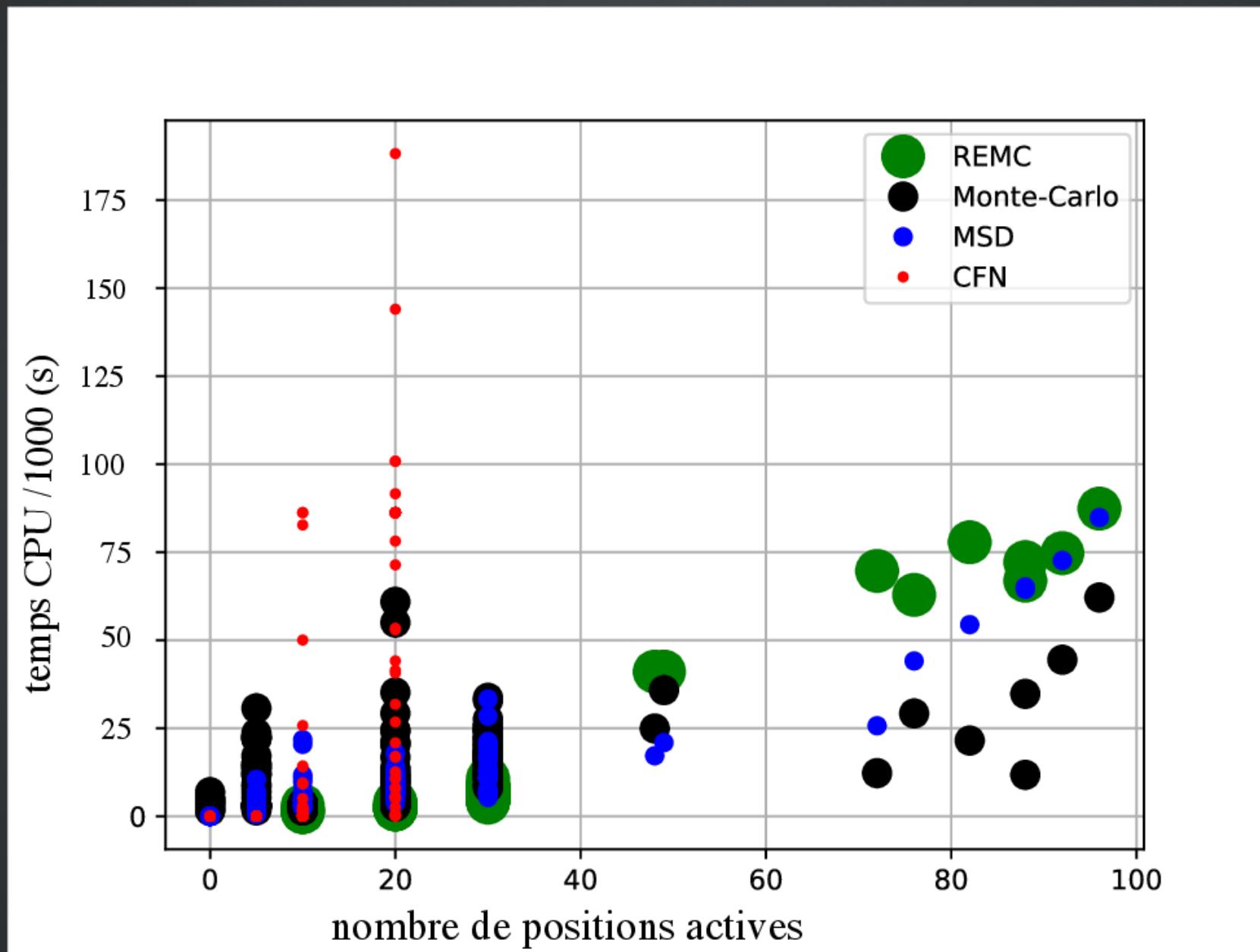
- Tests à 30, 20, 10 et 5 positions actives
- Dans chaque cas 5 sélections en privilégiant les ensembles en interaction pour chacune des 9 protéines.

Résultats

Différences avec le GMEC

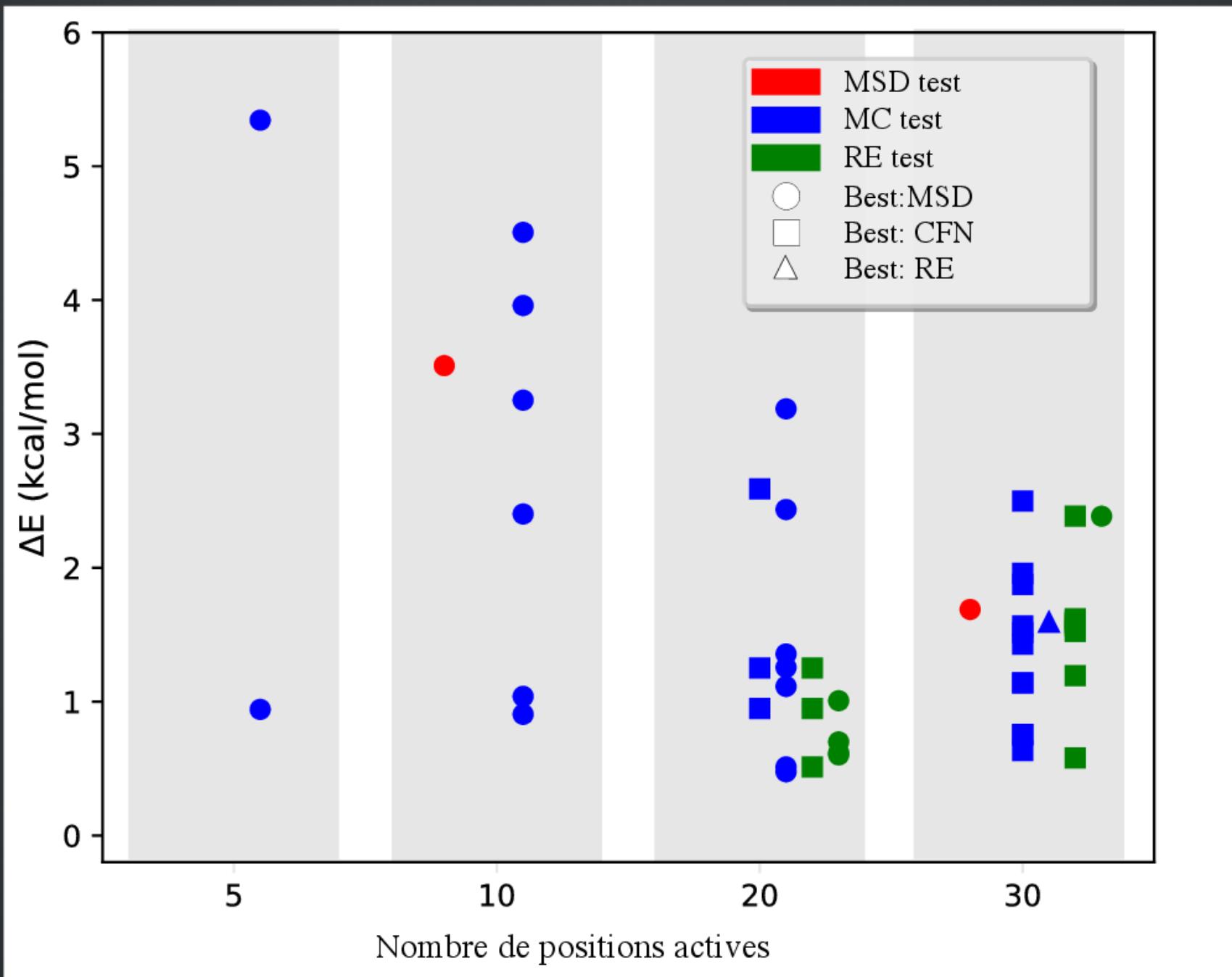


Temps CPU



Résultats

Différences avec la meilleure énergie commue



Études de quelques cas

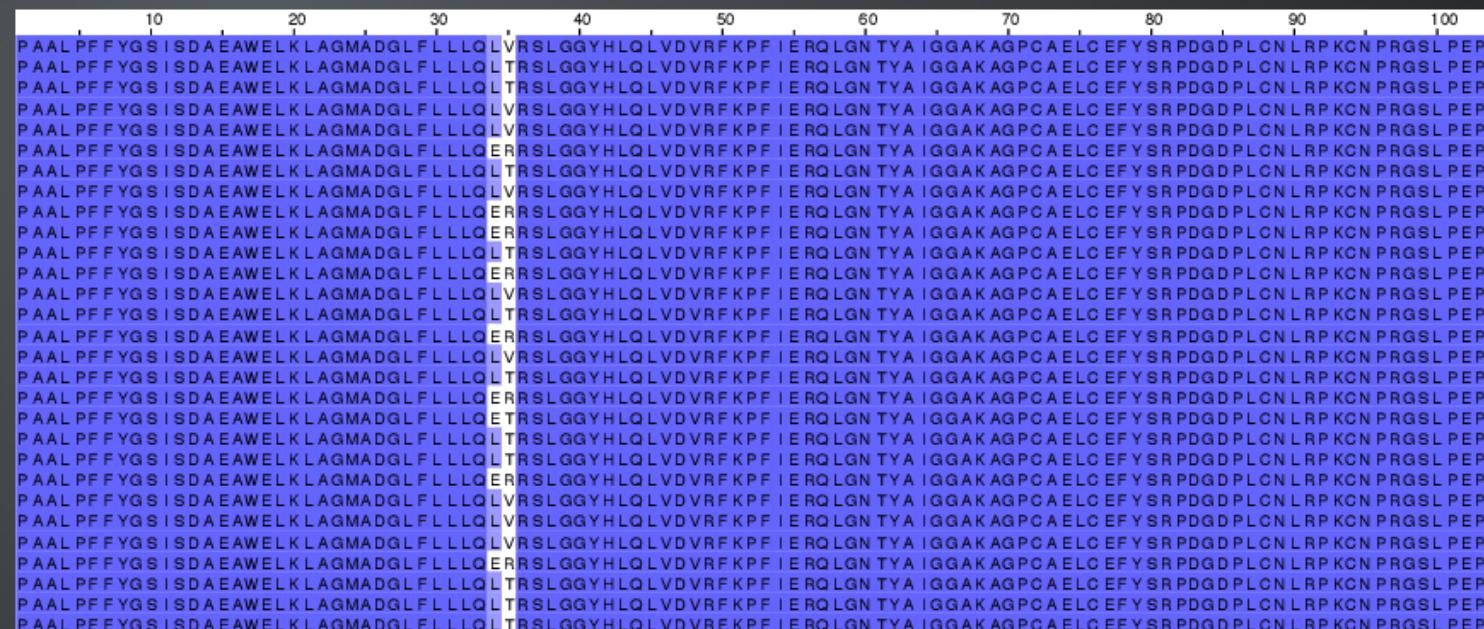
Séquences au voisinage du GMEC

1CKA 10 actifs

1M61 10 actifs



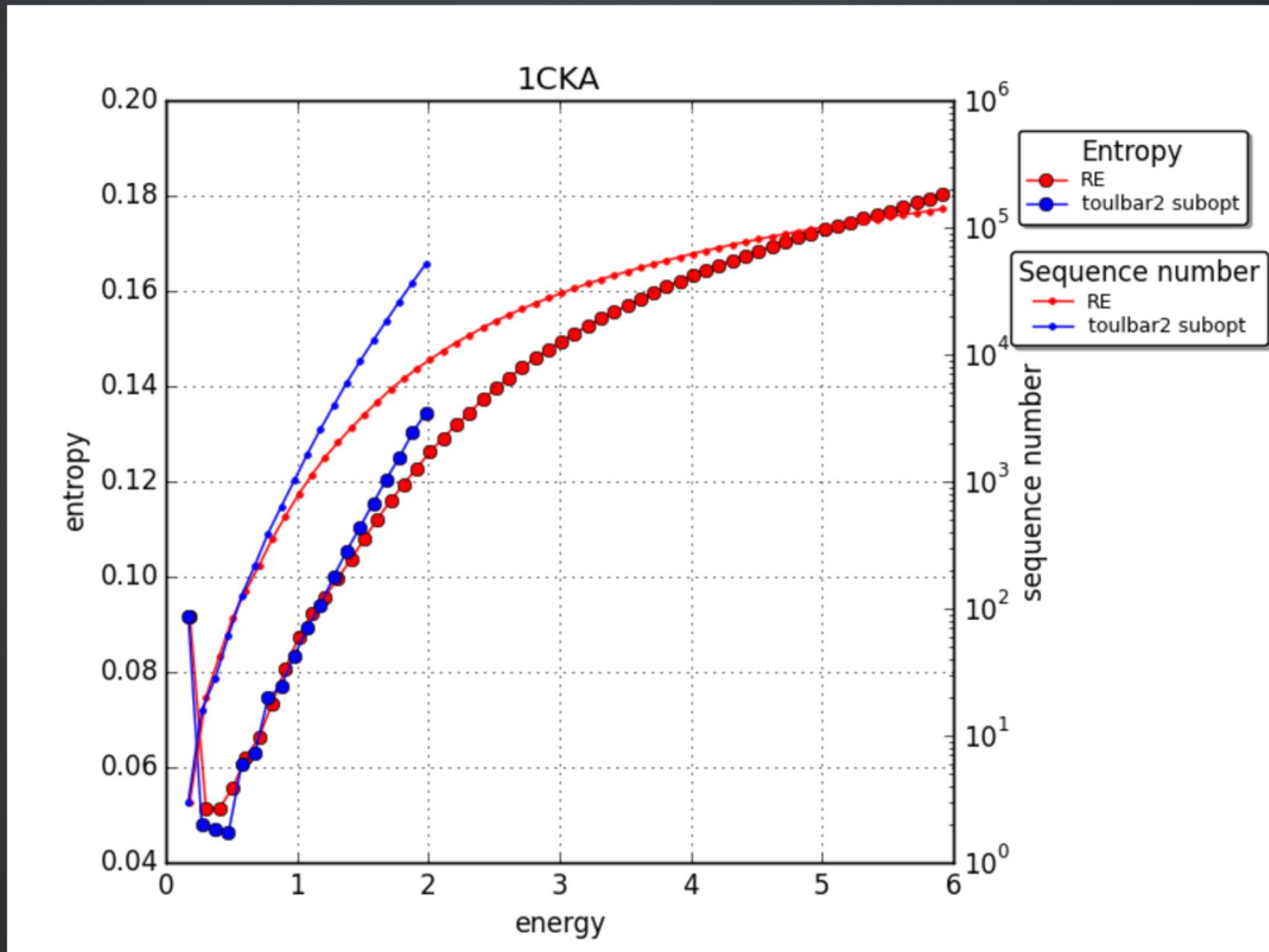
Difficile pour le Monte Carlo



Difficile pour le MSD

Études de quelques cas

densité au voisinage du GMEC



Mise à jour Proteus

Optimisation du Monte Carlo

- Ajustement du facteur Hasting
- critère sur la meilleure énergie
- changement de la construction d'un pas

Notion de rotamère dans Xplor

Nouvelle méthode de calcul des énergies de références

Optimisations des énergies de références

L'énergie de l'état déplié d'une séquence S est de la forme:

$$E_s^u = \sum_{i \in s} E_{t_i}^r$$



Ce sont des paramètres ajustables

L'objectif est de reproduire des fréquences d'acides aminés cibles.

Les homologues naturelles donnent les fréquences cibles.

3 algorithmes sont utilisés:

- Une méthode d'ajustement logarithmique:

$$E_t^r(n+1) = E_t^r(n) + c \ln(freq_t^{MC}(n)/freq_t^{naturelle})$$

- La méthode du gradient
- Une méthode du gradient à pas variable

Maximiser la vraisemblance des énergies de références par la méthode du gradient

Soit \mathcal{S} un ensemble de séquences naturelles , $P(\mathcal{S})$ sa probabilité de Boltzmann est une fonction des E_t^r .

Nous cherchons les E_t^r qui maximisent $P(\mathcal{S})$, elles réalisent notre objectif.

La méthode consiste à avancer dans la direction du gradient de $\ln(P(\mathcal{S}))$

On a $\frac{1}{N} \frac{\partial}{\partial E_t^r} \ln P(\mathcal{S}) = freq_t^{\mathcal{S}} - \langle n(t) \rangle$

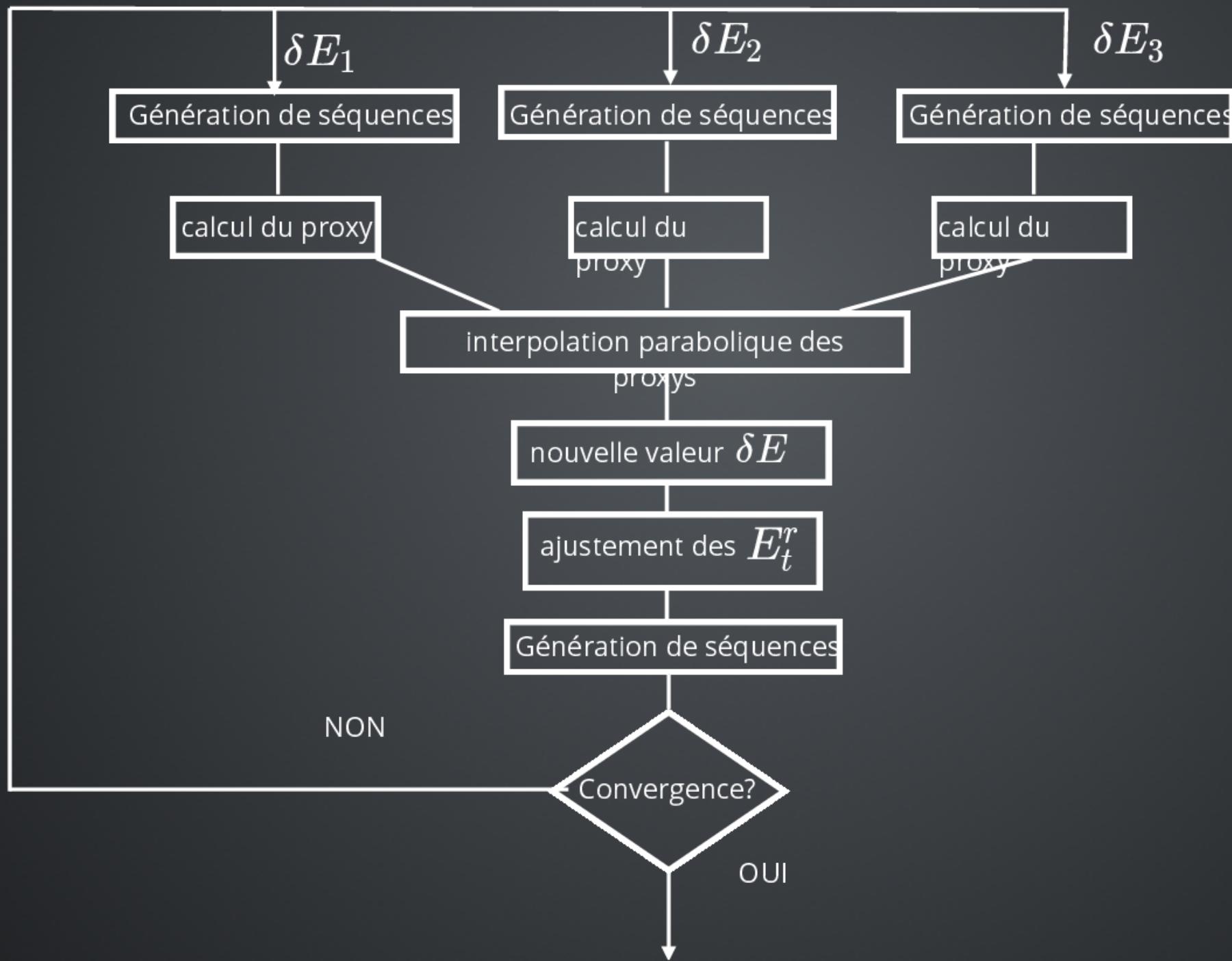
L'algorithme itératif:

$$E_t^r(n+1) = E_t^r(n) + \delta E \times (freq_t^{\mathcal{S}} - freq_t^{MC}(n))$$

La méthode du gradient à δE variable

Une fonction proxy C comme outil de mesure de l'état de l'optimisation

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



Une nouvelle approximation GB

Le "Fluctuating Dielectric Boundary" (FDB)

Le terme GB est une somme sur les paires d'atomes de la protéine:

$$E_{GB} = \sum_{i,j}^N g_{ij}(b_i, b_j)$$

On introduit un rayon de solvatation moyen d'un résidu I :

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

Ce qui permet d'avoir:

$$E_{GB} = \sum_{I,J}^N G(B) \quad \text{avec} \quad B = B_I B_J$$

G est fonction de la position relative des chaînes latérales.

On a:

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

Les c_k^{IJ} peuvent être stockés dans la matrice.

Dessin computationnel de domaines PDZ

les protéines

les homologues

	nombre de positions actives	nombre	E-value	% identité
NHREF	76	62	< 1e-32	67-95
Syntenine	72	85	< 1e-43	85-95
DGL2	82	43	< 1e-41	78-95

le protocole

Les matrices sont calculées avec MM/GB/SA.

GLY et PRO sont fixés

Les Eref avec la méthode du gradient à pas variable:

- optimisation sur des classes d'acides aminés
- puis sur les types.
- 2 jeux d'énergies selon la position enfui/exposée
- Les deux variantes GB (NEA et FDB)

Le meilleur protocole REMC

Sélection des 10000 meilleures séquences

Caractéristion des séquences calculées:

Superfamily, similarité Pfam, entropie résiduelle

Comparaison avec Rosetta (fixbb)

Résultats Superfamily et identité

Proteus

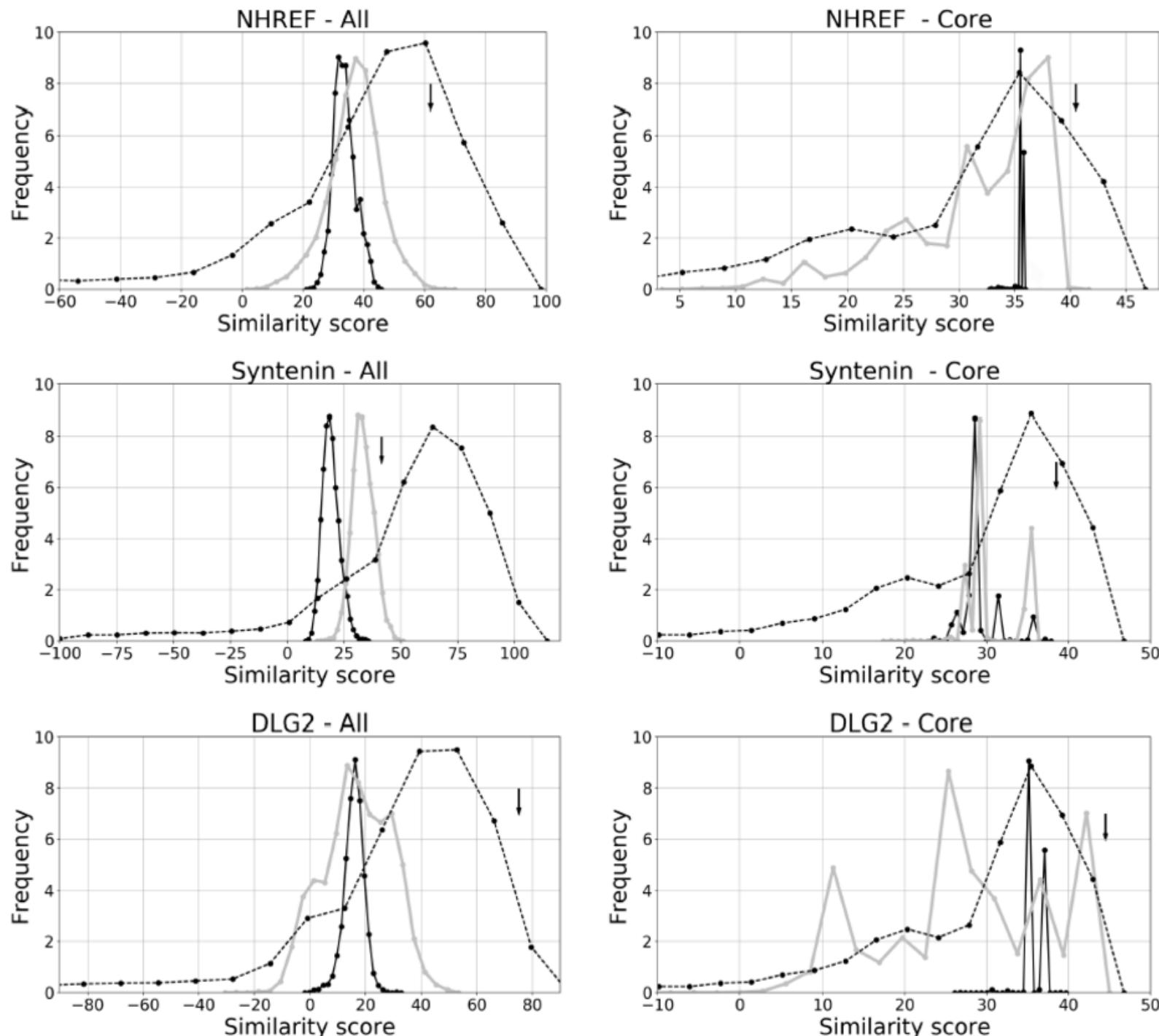
Protéine	identité à la native	E-value famille	Super-famille	succès Super-famille	E-value famille	succès famille
NHREF	24%	8,54 10 ⁻¹⁴		100%	8,94 10 ⁻³	100%
Syntenine	31%	2,85 10 ⁻⁶		100%	2,69 10 ⁻³	100%
DLG2	33%	3,26 10 ⁻¹²		100%	1,96 10 ⁻³	100%

Rosetta

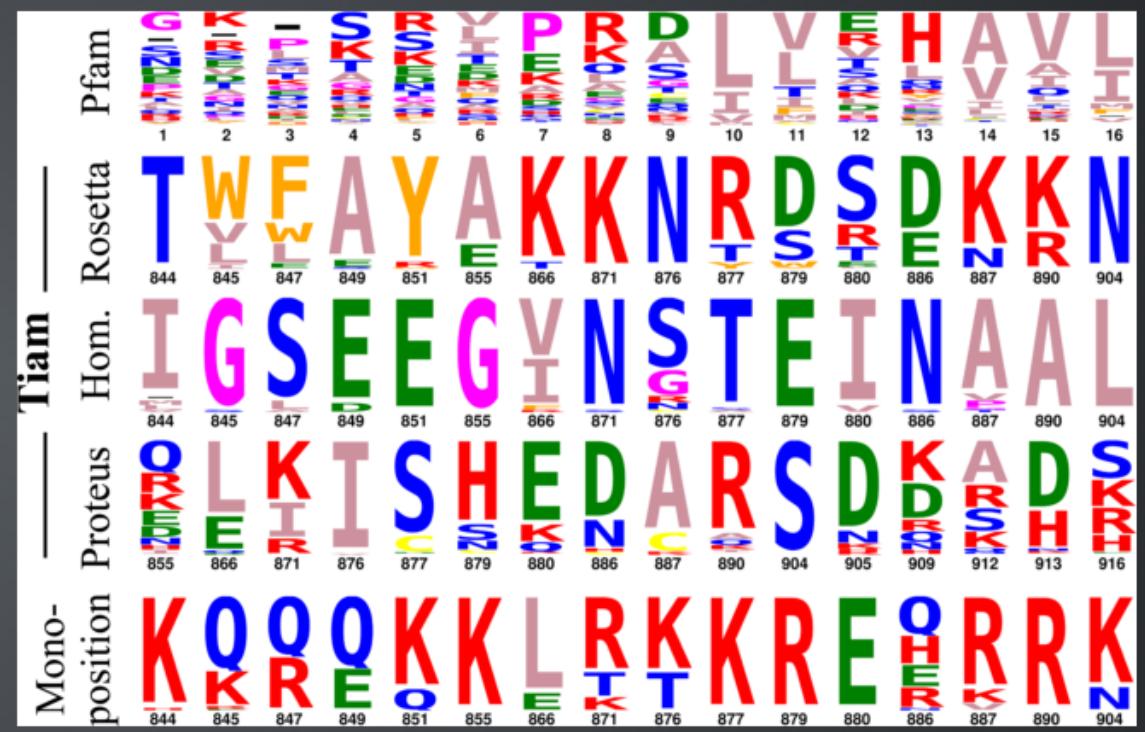
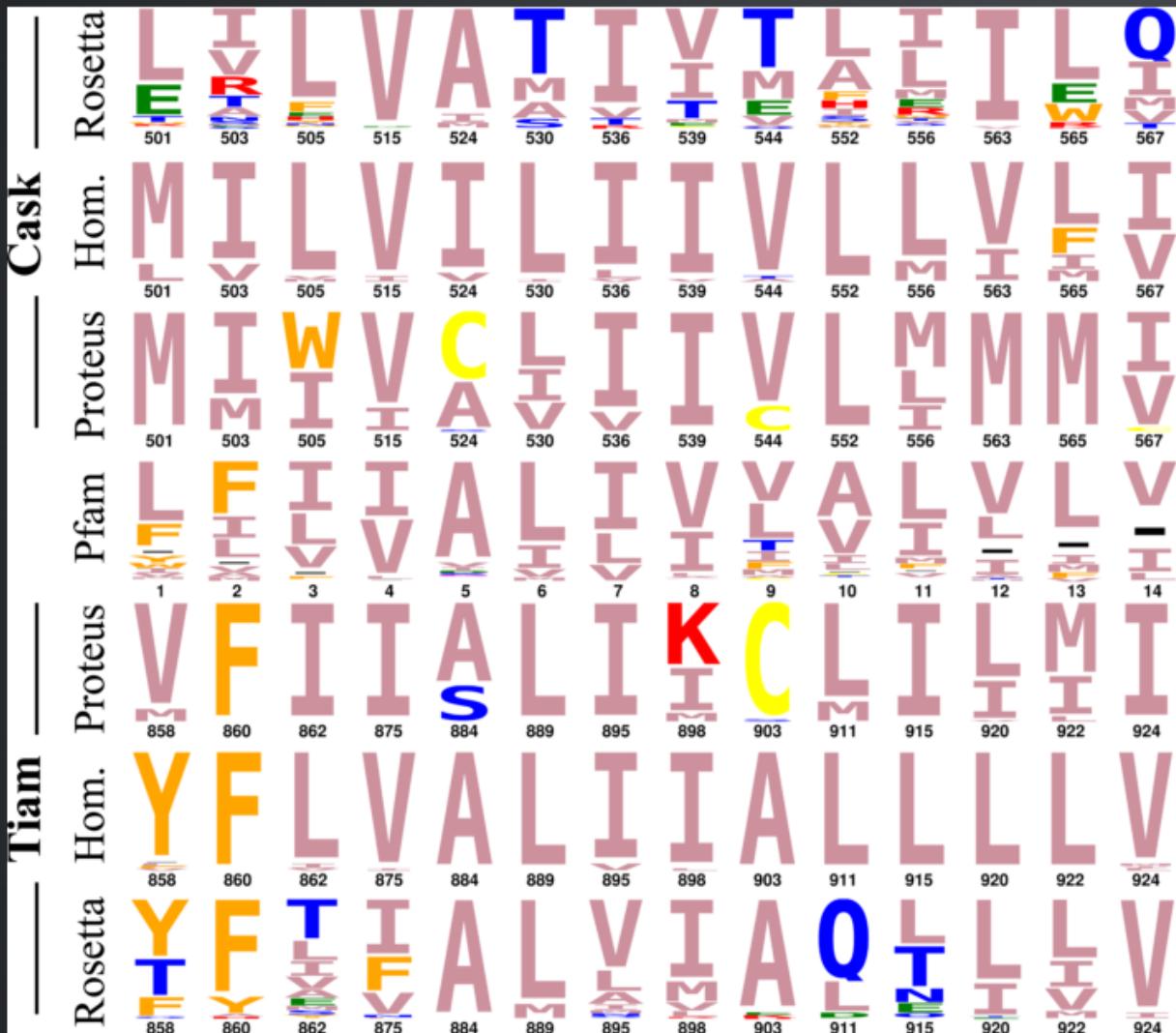
Protéine	identité à la native	E-value famille	Super-famille	succès Super-famille	E-value famille	succès famille
NHREF	35%	1,3 10 ⁻¹³		100%	2,2 10 ⁻³	100%
Syntenine	38%	7,3 10 ⁻¹³		100%	1,8 10 ⁻³	100%
DLG2	40%	1,3 10 ⁻⁹		100%	9,6 10 ⁻⁴	100%

Similarité

- Proteus
- Rosetta
- Pfam
- Native



Séquence Proteus et Rosetta sous forme de logos



Entropie

Protéine	Proteus	Rosetta	Pfam "seed"
NHREF	1,38	1,45	3,15
INAD	1,37	1,55	3,06
GRIP	1,33	1,44	3,06
Syntenin	1,39	1,43	3,03
DLG2	1,24	1,57	3,11
PSD95	1,27	1,40	3,15
6 protéines	2,42	2,88	
Cask	1,55	1,65	3,15
Tiam1	1,22	1,57	3,15

Applicaiton: Croissance du noyau hydrophobe

Possibilité de "design" du cœur hydrophobe des domaines PDZ

Tiam1 et Cask sont soumis à plusieurs simulations Proteus.

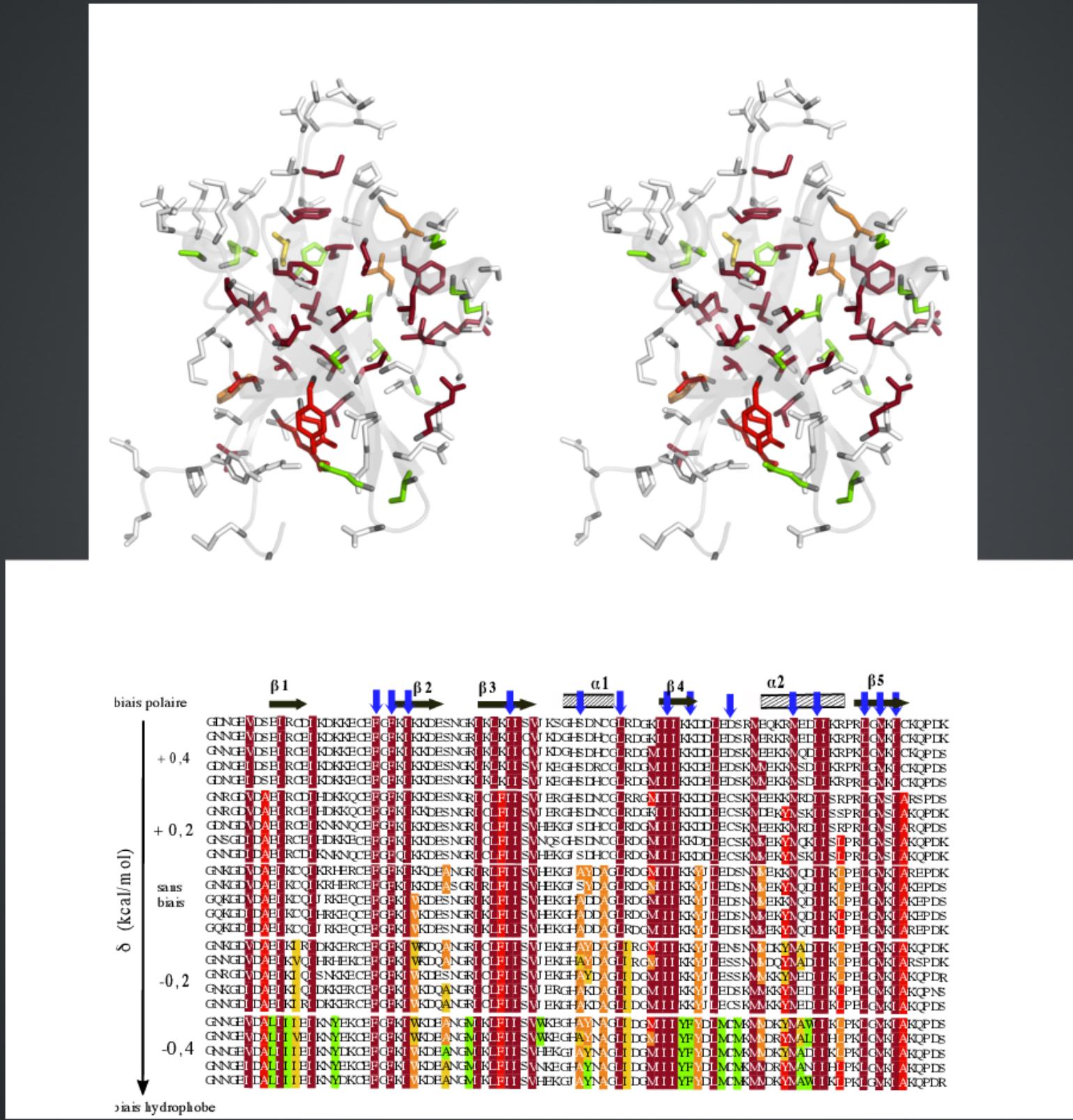
Les simulation sont biaisées graduellement , via les Eref.

Les types hydrophobes sont pénalisés au début , puis progressivement favorisés.

Introduction d'un indice de changement de type d'acide aminé par unité d'énergie du biais:

$$\psi_h = \frac{1}{N} \frac{\delta N}{\delta E}$$

Croissance du noyau hydrophobe



Conclusion