Apollinaire Roubert

Ilyas Grandguillaume

Conception d’un programme de threading par double programmation dynamique.

Contexte

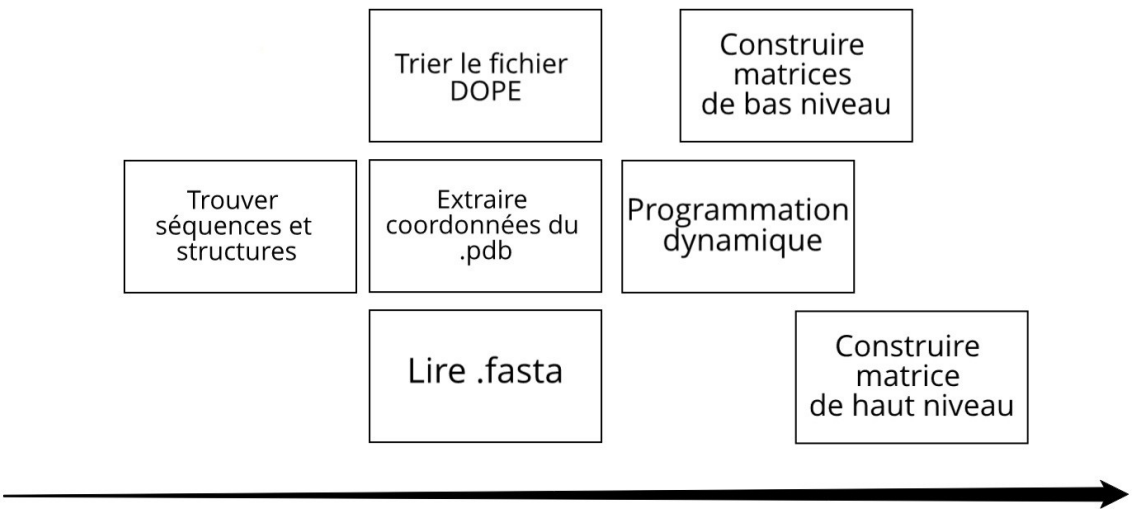
La fonction d’une protéine est intrinsèquement liée à sa structure tri-dimensionnelle, cependant avec la découverte de plus en plus de nouvelles séquences protéiques, les méthodes de résolution de structure sont inaptes à toutes les résoudre. Tandis que la taille des bases de données augmente, de nouvelles méthodes de résolution émergent pour pouvoir répondre à cette nouvelle problématique. La première et la plus performante est la méthode d’association par homologie de séquence, mais certaines séquences ne comportent pas un taux d’homologie assez important pour utiliser cette dernière. Il est dans ce cas possible de faire recours aux méthodes de threading, qui reposent sur l’alignement entre une séquence et des positions d’une structure déjà résolue.

Les méthodes de threading nécessitent une séquence d’intérêt que l’on enfile sur une librairie de structure 3D. Ensuite on évalue la vraisemblance de l’alignement séquence – structure.

Dans ce projet on s’intéresse spécifiquement à la manière dont on enfile la séquence sur la structure.

Organisation et démarche

L’organisation du développement du programme s’est faite comme suit :



Pour parvenir à enfiler une séquence sur une structure donnée on utilise une technique de double programmation dynamique. Pour cela on procède en deux étapes : tout d'abord on créée une collection de matrices de bas niveau, chaque matrice correspond à un acide aminé de la séquence d'intérêt fixé à une position de la structure 3D. Après avoir fixé la contrainte « acide aminé position », on place chaque autre acide aminé de la séquence d'intérêt de manière à ce qu'il maximise le score de notre contrainte.

Une fois les matrices de bas niveaux remplies, on peut passer à la construction de la matrice de haut niveau finale par programmation dynamique classique. La particularité de la matrice finale est que l'on utilise les score finaux des matrices de bas niveau pour la remplir. Pour finir on retrace un chemin optimal sur cette dernière qui nous donnera l'association séquence – position.

Matériel et méthodes

Les ressources utilisées lors de ce projet sont peu nombreuses : elles comportent le fichier .pdb correspondant à la structure ; le fichier .fasta dans lequel se trouve la séquence d'intérêt ; Python (version 3.6.9) et certains de ses modules dont Biopython, numpy et pandas ; et finalement un fichier contenant les potentiels statistiques DOPE, qui donnent un score associé à la distance entre deux atomes pour chaque paire de résidus en résumant son environnement.

Nous avons choisi la structure de 1AXH qui est une neurotoxine d'araignée comme template et 1RKK (protéine antimicrobienne synthétique) comme séquence cible pour tester notre algorithme car les deux séquences sont très similaires. Ensuite nous avons choisi une autre séquence 6PI2 avec un motif de structure secondaire en commun (bêta hairpin) pour mettre notre algorithme à l'épreuve.

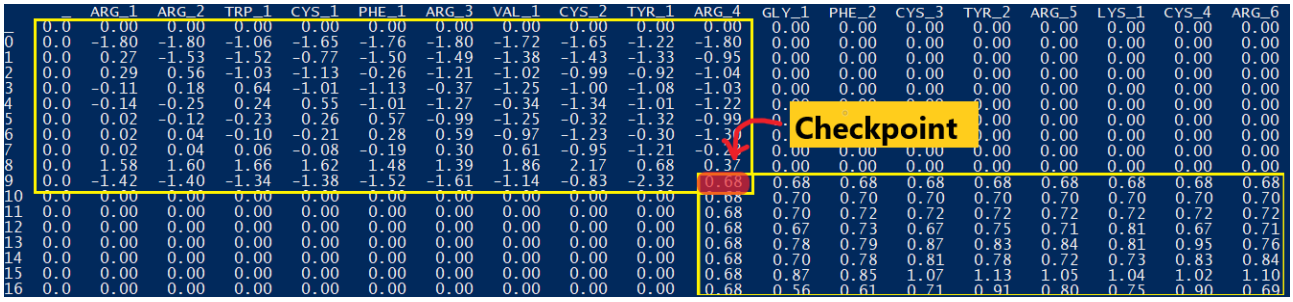
Résultats et discussion

Nous avons implémenté une version simplifiée du programme Threader, dans cette partie nous discuterons des choix algorithmiques que nous avons fait, ainsi que de nos alignements finaux.

Le seul objet que l'on a implémenté représente la matrice de programmation dynamique. Ceci a été décidé car le programme demande la création d'un très grand nombre de ces dernières pour les matrices de bas niveau, mais aussi pour la matrice de haut niveau qui est elle-même une matrice de programmation dynamique. Afin de contenir toutes les méthodes nécessaires, notamment le remplissage de cette matrice selon si c'est une matrice de bas ou haut niveau, la classe est devenu très volumineuse. Il aurait été possible de créer 2 autres classes pour les 2 niveaux de matrices qui auraient héritées d'une classe plus générale, mais par un manque de temps ceci n'a pas été implémenté.

Rétrospectivement, nous aurions du garder dans un objet séquence toutes les informations concernant la séquence d'intérêt ainsi que les scores dope d'appariement entre les différents acides aminés de cette dernière. Ceci aurait permis de clarifier

Tout d'abord, pour construire les matrices de bas niveaux, il a fallu fixer un acide aminé à une position. Nous avons procédé par la mise en place d'un checkpoint (contrainte acide aminé – position), et forcé le passage par ce point de la matrice. La résolution d'une matrice de bas niveau s'est faite en deux partie : du point de départ [0,0] en haut à gauche jusqu'au checkpoint, puis du checkpoint au point d'arrivée [x,y] en bas à droite. (



	_	ARG_1	ARG_2	TRP_1	CYS_1	PHE_1	ARG_3	VAL_1	CYS_2	TYR_1	ARG_4	GLY_1	PHE_2	CYS_3	TYR_2	ARG_5	LYS_1	CYS_4	ARG_6
_	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0	0.0	-1.80	-1.80	-1.06	-1.65	-1.76	-1.80	-1.72	-1.65	-1.22	-1.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	0.0	0.27	-1.53	-1.52	-0.77	-1.50	-1.49	-1.38	-1.43	-1.33	-0.95	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.0	0.29	0.56	-1.03	-1.13	-0.26	-1.21	-1.02	-0.99	-0.92	-1.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
3	0.0	-0.11	0.18	0.64	-1.01	-1.13	-0.37	-1.25	-1.00	-1.08	-1.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4	0.0	-0.14	-0.25	0.24	0.55	-1.01	-1.27	-0.34	-1.34	-1.01	-1.22	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
5	0.0	0.02	-0.12	-0.23	0.26	0.57	-0.99	-1.25	-0.32	-1.32	-0.99	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6	0.0	0.02	0.04	-0.10	-0.21	0.28	0.59	-0.97	-1.23	-0.30	-1.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
7	0.0	0.02	0.04	0.06	-0.08	-0.19	0.30	0.61	-0.95	-1.21	-0.2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	0.0	1.58	1.60	1.66	1.62	1.48	1.39	1.86	2.17	0.68	0.37	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	0.0	-1.42	-1.40	-1.34	-1.38	-1.52	-1.61	-1.14	-0.83	-2.32	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68
10	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.68	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70
11	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.68	0.70	0.72	0.72	0.72	0.72	0.72	0.72	0.72
12	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.68	0.67	0.73	0.67	0.75	0.71	0.81	0.67	0.71
13	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.68	0.78	0.79	0.87	0.83	0.84	0.81	0.95	0.76
14	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.68	0.70	0.78	0.81	0.78	0.72	0.73	0.83	0.84
15	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.68	0.87	0.85	1.07	1.13	1.05	1.04	1.02	1.10
16	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.68	0.56	0.61	0.71	0.91	0.80	0.75	0.90	0.69

Figure 1: Matrice bas niveau avec la contrainte Arg_4 en position 9

Une autre possibilité aurait été de partir de notre contrainte et de progresser dans les deux sens pour résoudre la matrice, mais cette alternative s‘est avérée beaucoup plus complexe que la première et n’a pas été implémentée.

Après construction des matrices de haut niveau, une partie de notre algorithme remonte un des chemin optimal et affiche l’alignement séquence – position sur le terminal. Il aurait été possible de créer un fichier pdb à partir des positions pour visualiser l’alignement mais il n’aurait pu inclure qu’une suite de carbones alphas car il aurait été difficile de replacer les positions des atomes du reste des résidus.

La séquence de 1RKK et la structure de 1AXH se sont bien alignées (voir annexe), et ceci sur le brin bêta final du motif bêta hairpin de la structure (positions 28 à 37 complètement enfilées avec les acides aminés terminaux). Nous nous attendions à un alignement des deux brins du bêta hairpin, mais c’est probablement la petite taille du feuillet bêta de 1RKK ou la présence d’une torsion dans le motif de 1AXH qui à empêché un alignement complet.

Pour la séquence de 6PI2 et la structure de 1AXH l’alignement était très similaire avec le précédent (voir annexe).

A l’inverse la structure de 6PI2 et la séquence de 1RKK (qui sont homologues) ne s’alignent pas correctement : on obtient une alternance de gap de sorte qu’il n’y a presque aucun alignement (voir annexe).

Nous avons remarqué que les longues séquences s‘alignait mal sur les petites structures, et qu’a l’inverse les petites séquences s’alignent bien sur les grandes structures. Ceci est peut être du au fait que l’on a pénalisé davantage les gaps (conseillé : 0 , utilisé -2), cependant sans cette pénalité l’algorithme n’arrivait pas à enfiler une séquence sur une structure (même identique).

Il aurait été possible d’implémenter une pénalité de gap consécutifs qui aurait peut être amorti ce problème de taille de structure vs séquence, mais par manque de temps ceci a été écarté.