2022 - 2023

Projet court : Calcul de la surface accessible au solvant d'une protéine.

SUJET N°2 / M2-BI

MALASSIGNE VICTOR

Université Paris Cité

Introduction

Dans un article publié en 1973, Shrake, A et Rupley, JA calculent la surface d'une protéine accessible au solvant. Dans le cadre de ce projet, nous voulons reprogrammer cette modélisation dans un script en langage python. Pour ce faire, nous sommes partis du principe de faire « rouler » une molécule d'eau à la surface des atomes d'une protéine. Pour cela, nous modélisons sur chaque atome une sphère avec le rayon de Van der Waals de l'atome (Figure 1). Sur ces sphères, nous ajoutons une molécule d'eau (représentée par son rayon de Van der Waals 1,7Å) que nous déplaçons point par point de ladite sphère. Tant que la molécule d'eau passe sans recouvrir deux sphères cela signifie que la surface est exposée au solvant.

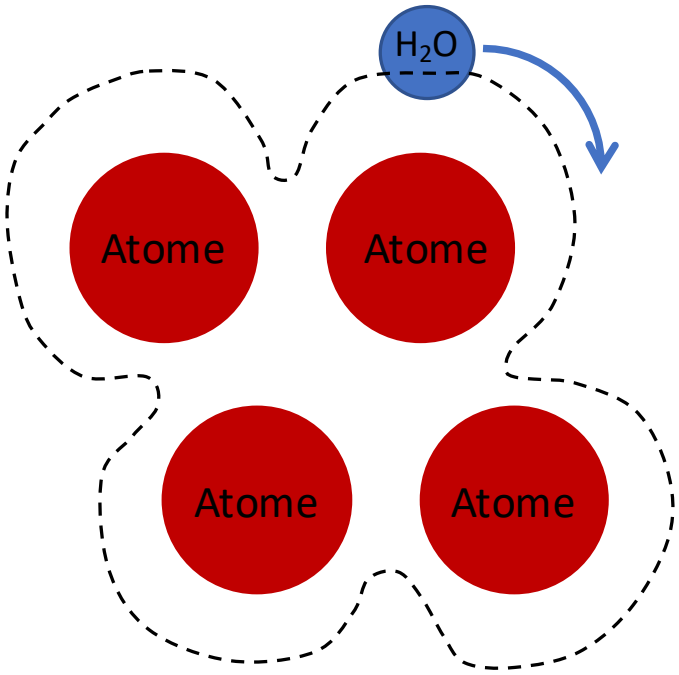

FIGURE 1 : SCHEMA DE LA SURFACE ACCESSIBLE AU SOLVANT.

En se basant sur ce principe, il est possible de déterminer la surface accessible au solvant d'une protéine. Les résultats de notre algorithme seront ensuite comparés avec les résultats de DSSP (*Define Secondary Structure of Proteins*) utilisé classiquement.

Matériel

- Python : ce programme a été codé et compilé avec la version 3.10.4 de python
- Conda : nous utilisons la version 4.12.0 de conda pour créer un environnement disponible sur Github. Nous utilisons les modules suivants : Numpy, Bio.PDB, Scipy et sys.
- Github : un répertoire sur Github a été créé et est disponible sur : https://github.com/vmalass/2022_M2-BI_Projet_court.git. L'utilisation de l'outil Git et du service d'hébergement Github permet un archivage des différentes scripts et la sauvegarde des modifications avec la chronologie.

- Données : les protéines utilisées ont été téléchargées depuis la banque de données de protéines PDB (*Protein Data Bank*, https://www.rcsb.org/). Nous utilisons notamment la protéine 3i40.pdb (*Human insulin*) dans notre exemple. D'autre protéines ont été testées (8dev, 8d23, 7r4h, 7pbp, 7u52 et 6a5j), cela nous permet d'avoir un large éventail de tailles de protéine pour ensuite comparer nos résultats.
- DSSP : nous utilisons le logiciel DSSP qui est un algorithme permettant en partie ,de calculer surface accessible au solvant d'une protéine.

Méthodes

1. A partir d'un fichier .pdb nous extrayions les coordonnées, l'identité et le résidu parent de chaque atome avec un *parser* disponible dans le module Bio.PDB.
2. Nous calculons la distance entre chaque atome à l'aide du module scipy. ce qui nous génère une matrice de distances de dimension : nombre d'atomes × nombre d'atomes.
3. Nous recherchons tous les voisins de l'atome étudié dans un rayon de 10Å à partir de la matrice de distance.
4. Création d'une sphère autour d'un atome. La sphère se compose de 92 points répartis uniformément autour du centre de l'atome. Les points sont situés à une distance équivalente au rayon de Van der Waals de l'atome avec l'ajout du rayon de Van der Waals du solvant qui est l'eau.
5. Calcule de la distance euclidienne entre un point de la sphère et l'atome voisin.
6. Condition de sélection : si la distance entre le point de la sphère et l'atome voisin est inférieure au rayon de Van der Waals de l'atome ajouté au rayon de l'eau alors le point n'est pas accessible au solvant. S'il est supérieur, alors il est accessible.
7. Nous regardons tous les voisins par point de la sphère si un voisin rend le point inaccessible alors on passe au point suivant.
8. Nous calculons la surface accessible de l'atome avec le calcule suivant : $\frac{nombre\ de\ points\ accessibles}{nombre\ de\ points\ de\ la\ sphère} \times (4 \times \pi \times (rayon\ atome\ voisin + rayon\ de\ l'eau)^2$
9. Nous calculons de l'aire accessible au solvant par tous les atomes en faisant la somme de l'étape 8.

10. Nous calculons de la surface accessible relative de la protéine.
11. Nous calculons du pourcentage d'accessibilité au solvant.
12. Nous comparons les sorties avec DSSP.

Résultats

Au total, 7 protéines (voir Matériel) sont étudiées. Ces protéines varient de 13 à 1 048 résidus pour pouvoir tester la robustesse de l'algorithme sur un panel varié. Le premier point est de constater que pour les petites protéines l'algorithme prend quelques secondes pour sortir les résultats tandis que pour les plus grosses cela se compte en minute voire plusieurs minutes.

Nous montrons un tableau récapitulatif avec notre programme SASCM (*Solvent Accessible Surface Calculation Model*) et le programme DSSP. Nous calculons l'écart-type ainsi que le pourcentage d'erreur de notre programme (Tableau1).

Protéine	SASCM	DSSP	Ecart-type	% erreur
6a5j	1710,6	1556,0	109,3	9,9%
3i40	3297,4	3437,0	98,7	4,1%
8d23	18335,8	18966,0	445,6	3,3%
7u52	33308,8	34543,0	872,7	3,6%
7r4h	75704,1	80039,0	3065,3	5,4%
7pbp	81980,0	91019,0	6391,5	9,9%
8dev	119617,3	128561,0	6324,2	7,0%

TABLEAU 1 : RECAPITULATIF DES RESULTATS ET ANALYSES.

Dans la figure 2-A nous comparons les sorties de l'algorithme DSSP et SASCM pour la surface accessible au solvant de chaque protéine. Les barres d'erreurs représentent l'écart-type (Tableau 1). Nous constatons que pour chaque protéine, les résultats sont assez proches. Dans la figure 2-B nous étudions la corrélation entre les deux méthodes. Nous remarquons que les protéines se retrouvent pratiquement sur la même droite et que le R^2 est de 0,99 ce qui traduit une forte corrélation entre les deux méthodes.

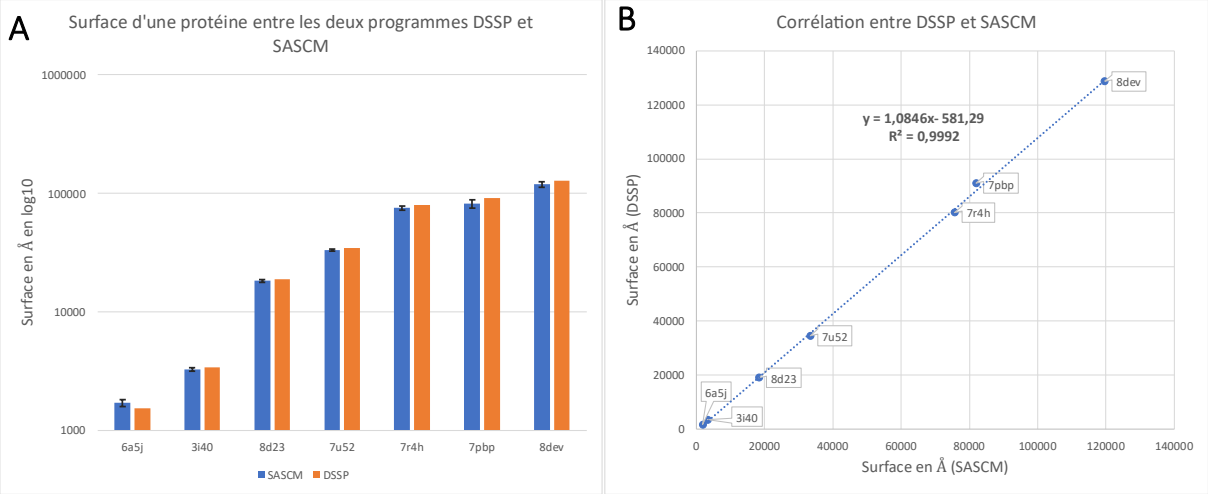

FIGURE 2 : DIAGRAMME EN BAR DE LA SURFACE DES PROTEINES ACCESSIBLES AU SOLVANT ET DROITE DE CORRELATION ENTRE DSSP ET SASCM

Conclusion

Avec toutes ces informations, nous pouvons conclure que notre algorithme SASCM obtient des résultats proches de DSSP avec une marge d'erreur variant de 3 à 10 %. La principale différence reste dans la rapidité d'exécution du programme. Une amélioration peut être apportée avec une parallélisation du code ou l'utilisation d'OOP (*Object Oriented Programming*).

Des difficultés sont survenues au début de la programmation avec l'utilisation du module pandas qui permet de générer des *data frames*, un abandon de cette méthode au profit de numpy qui permet de générer des *arrays* a été décidé.

Référence

Shrake A, Rupley JA. Environment and exposure to solvent of protein atoms. Lysozyme and insulin. J Mol Biol. 1973 Sep 15;79(2):351-71. doi: 10.1016/0022-2836(73)90011-9. PMID: 4760134.