2022 - 2023

Projet court : Calcul de la surface accessible aux solvants d’une protéine.

Sujet n°2 / M2-BI

Malassigne victor

# Introduction

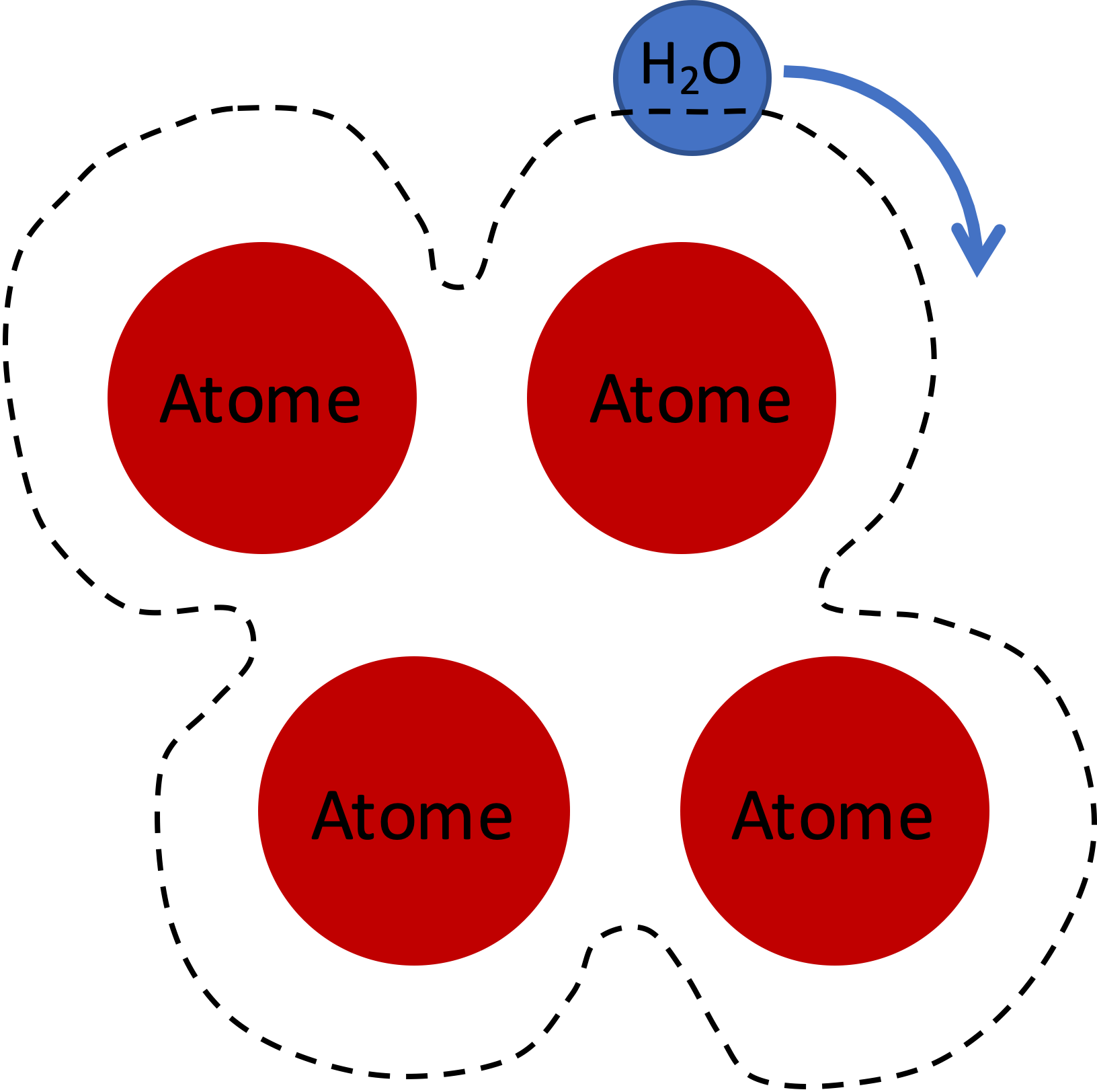
 Dans un article publié en 1973 de Shrake, A et Rupley, JA calcule la surface d’une protéine accessible aux solvant. Dans le cadre de ce projet nous voulons recoder cette modélisation en script python. Pour ce faire nous somme parti du principe de faire « rouler » une molécule d’eau à la surface des atomes. Pour cela nous modélisons sur chaque atome une sphère représentant le rayon de Van der Waals (Figure 1). Sur ces sphères nous ajoutons une molécule d’eau (représenté par son rayon de Van der Waals 1,7Å) que nous déplaçons point par point de ladite sphère. Tant que le rayon de la molécule d’eau passe sans toucher deux sphères cela signifie que la surface est exposée aux solvants

Figure 1 : Schéma surface accessible aux solvants

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

# Matériel

* Python : ce programme a été codé et compilé avec une version de python 3.10.4
* Conda : nous utilisons la version 4.12.0 de conda pour créer un environnement disponible sur Github (<https://github.com/vmalass/2022_M2-BI_Projet_court.git>). Nous utilisons les modules suivants : Numpy, Bio.PDB, scipy et sys.
* Github : un répertoire sur Github a été utilisé disponible sur : <https://github.com/vmalass/2022_M2-BI_Projet_court.git>. L’utilisation de l’outil Git et sur service d’hébergement Github permet un archivage point par point des scripts, modification avec une chronologie.
* Données : les protéines utilisées ont été téléchargées sur <https://www.rcsb.org/> qui est une banque de données sur les protéines (PDB *Protein Data Bank*). Nous utilisons notamment la protéine 3i40.pdb (*Human insulin*) dans notre exemple. D’autre protéine ont été testé (8dev, 8d23, 7r4h, 7pbp, 7u52 et 6a5j), cela nous permet d’avoir un éventail de taille de protéine assez large pour comparer nos résultats.
* DSSP : nous utilisons le logiciel DSSP qui est un algorithme permettant de calculer entre autres la surface accessible aux solvants d’une protéine.

# Méthodes

1. A partir d’un fichier .pdb nous extrairons les coordonnées, l’identité et le résidu 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 distance de dimension nombre d’atome sur nombre d’atome.
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 à situer 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 inférieur au rayon de Van der Waals de l’atome ajouté au rayon de l’eau alors le point n’est pas accessible aux solvants 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 :
9. Nous calculons de l’aire accessible aux 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 aux solvants.
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 est de constaté que pour les petites protéines l’algorithme prend quelque seconde pour sortir les résultats tandis que pour les plus grosse cela se compte en minute.

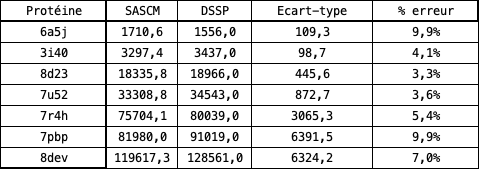
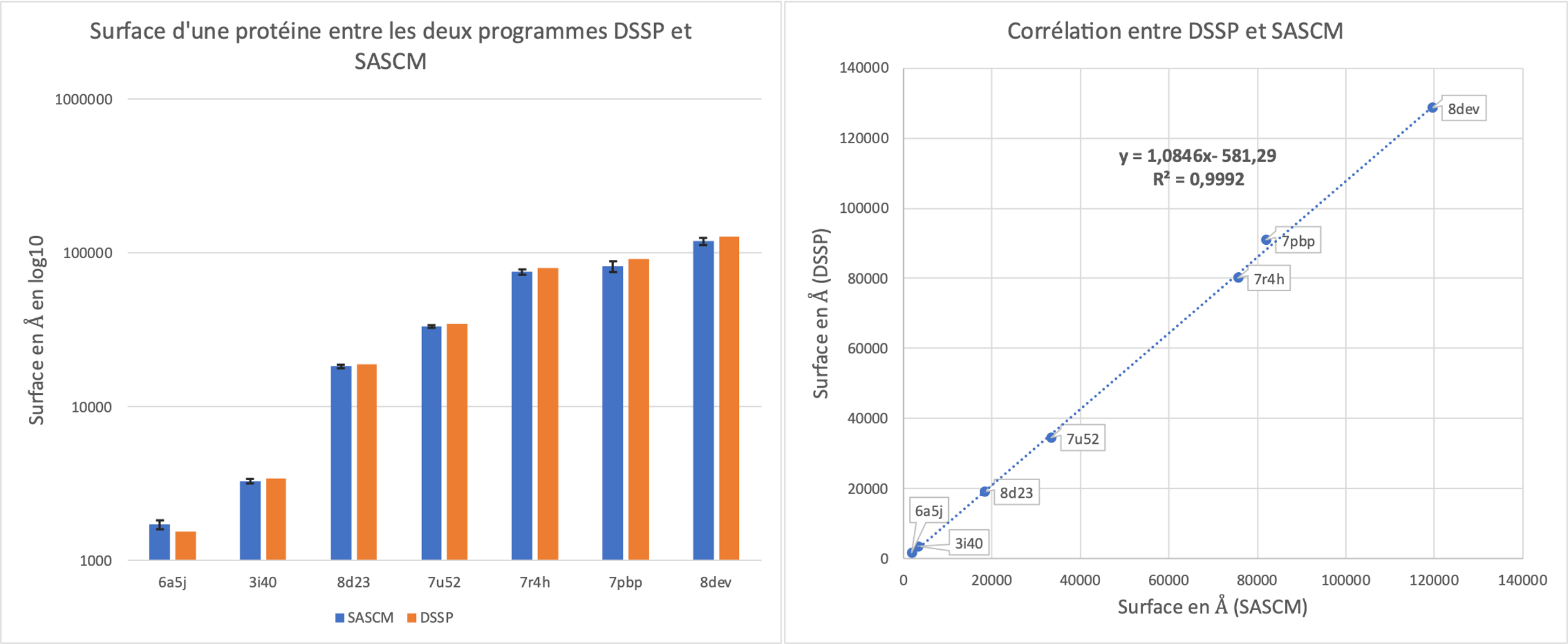
 Nous montrons un tableau récapitulatif avec notre programme SASCM (*Solvent Accessible Surface Calculation Model*) et le programme DSSP. Nous calculons l’écart type ainsi que le pourcentage d’erreur de notre programme (Tableau1).

Tableau 1 : Récapitulatif des résultats et analyses

Dans la figure 2-A nous comparons les sorties de l’algorithme DSSP et SASCM pour la surface accessible aux solvants de chaque protéine, les barres d’erreurs représente 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 retrouve pratiquement sur la même droite et que le R2 est de 0,99 ce qui traduit une forte corrélation entre les deux méthodes.



A

B

Figure 2 : Diagramme en bar de la surface des protéines accessible aux solvants et droite de corrélation entre DSSP et SASCM

# Conclusion

Avec toute 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 pourcents. La principale différence reste dans la rapidité d’exécution du programme. Une amélioration peut être apporté avec une parallélisation du code ou l’utilisation d’OOP (*Object Oriented Programming*).

Des difficultés sont survenu en début de programme 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 choisi.

# 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.