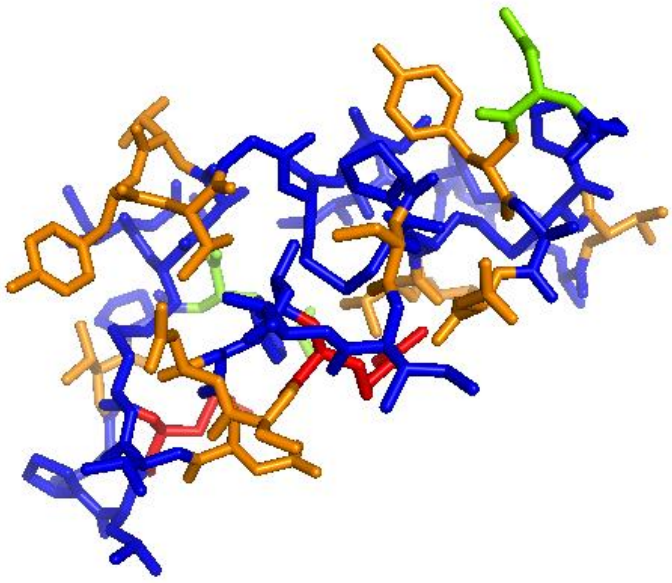
PROJET USI11 – Python 2 – GB4

Analyse d'une structure protéique au format PDB



Tom MAUGER-BIROCHEAU
Laura PRUGNEAU
Ophélie RABATEL
Génie Biologique – 4ème année

Romain GAUTIER
Programmation Python et extraction d'informations biologiques
Année universitaire 2021-2022

Table des matières

I. CONTEXTE BIOLOGIQUE

Suite au fort développement des techniques de séquençage avec notamment le Next Generation Sequencing, la quantité de données générée est devenue très importante. Par conséquent des questions de stockage et d'analyse se sont posées. Pour répondre à cette dernière, la bioinformatique est devenue un outil indispensable afin d'aider à l'analyse des séquences.

L'analyse des séquences ainsi que des structures protéiques permettent d'obtenir de nombreuses informations et propriétés biologiques de molécules du vivant. En effet, l'information génétique de tout organisme vivant est contenue dans une séquence. De plus, les séquences nucléotidiques et protéiques sont facilement manipulables informatiquement car ce sont des chaînes de caractères simples et fixes. Dans le cas de séquences nucléiques, il s'agit par exemple d'une répétition de 4 lettres identiques que sont les nucléotides, pour tous les organismes. Au vu de la répétitivité des séquences il est donc plus facile de créer des programmes aidant à l'analyse de ces séquences.

Dans les années 80, des bases de données sont créées afin de faciliter l'accès des différentes séquences et structures protéiques à la communauté scientifique. Cela permet donc d'obtenir différents fichiers sous forme d'un format identique ce qui facilite l'analyse de ces fichiers par un programme.

Par conséquent, la mise en place de programmes (même petits) permet d'aider amplement les chercheurs, en permettant un gain de temps mais également de limiter les erreurs d'analyse.

II. PRESENTATION GENERALE DU PROJET

L'objectif de ce projet est l'analyse d'une structure protéique au format PDB soit par l'ouverture d'un fichier directement disponible sur votre ordinateur soit par la récupération des données via le site web de la PDB. Différentes analyses seront ensuite effectuées sur ces données et seront stockées dans différents fichiers de résultats enregistrés sur votre ordinateur.

En partant d'un fichier au format PDB, l'utilisateur sera en mesure de visualiser un grand nombre d'informations sur cette séquence. Pour rappel, un fichier PDB (Protein Data Bank) est un fichier donnant des informations sur la structure tridimensionnelle de macromolécules biologiques.
Les différents éléments de sortie générés par notre programme sont :

- Une page internet permettant de visualiser le graphique d'hydrophobicité à l'aide de la méthode de Fauchere
- Une page internet permettant de visualiser le graphique d'hydrophobicité à l'aide de la méthode de Kyte
- Une page internet permettant de visualiser un graphique montrant la proportion des acides aminés de la protéine comparée aux proportions moyennes
- Une page internet permettant de visualiser la représentation *heat map* de la protéine
- Un fichier au format .fasta nommé « ''CODE_PDB''_FASTA.fasta » comportant la séquence Fasta de la protéine (informations sur la protéine + séquence)
- Un fichier au format .txt nommé « ''CODE_PDB''_output_file.txt » comportant la séquence protéique, la méthode expérimentale utilisée pour déterminer la structure, la résolution éventuelle (si cristallographie) et les ponts disulfures possibles avec la position des cystéines

concernées et leurs distances les unes par rapport aux autres ainsi que les cystéines non pontées.

- Un fichier au format .pdb nommé « ''CODE_PDB''_B_fact_modif.pdb » avec une modification des B-factors afin d'obtenir une visualisation des acides aminés par rapport aux propriétés physicochimiques des résidus composant la protéine.

Afin de faciliter les interactions avec l'utilisateur nous avons opté pour une interface graphique sous forme de menu. Cette interface sera détaillée plus précisément par la suite.

Dans la troisième partie de ce rapport, nous allons regarder plus en détails les différentes parties de ce programme et nous allons utiliser, comme exemple, le fichier PDB de la protéine 1CRN.

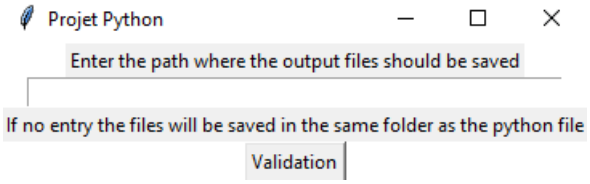
III. LA STRATEGIE DE PROGRAMMATION

1. Organisation du code

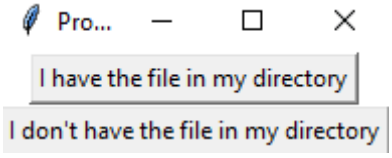
Pour permettre une bonne prise en main par l'utilisateur, nous expliquons les étapes préliminaires au lancement du programme :

Tout d'abord, il faut ouvrir le programme à l'aide d'un exécuteur de script Python. Les dépendances doivent être installées au préalable avant d'exécuter le code. Dans un deuxième temps, il est nécessaire d'installer les modules "Biopython" et "plotly". Pour cela, il suffit de les installer à l'aide d'une méthode pip ou conda. Par exemple, à l'aide de la méthode pip il faut écrire "pip install Biopython" dans la console de commande. Après l'installation des deux modules, le code pourra être exécuté correctement.

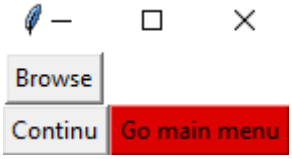
Lorsque le programme s'exécute, une fenêtre graphique apparaît. Il faut tout d'abord entrer le chemin d'accès dans lequel nous voulons que le fichier de sortie soit enregistré. Si aucun chemin d'accès n'est rentré par l'utilisateur, le fichier sera enregistré dans le même répertoire que le programme.



Puis une fenêtre pour savoir si l'on possède le fichier PDB dans notre répertoire apparaît. Si on clique sur "I have the file in my directory" cela va permettre d'ouvrir un fichier PDB présent sur votre disque dur. Alors que si l'on clique sur "I don't have the file in my directory" cela va permettre de récupérer le contenu d'un fichier au format PDB par l'interrogation du site web PDB avec le code PDB.



- Si on choisit de cliquer sur la première proposition, l'utilisateur va parcourir son répertoire pour trouver le fichier qu'il souhaite utiliser.



Puis en appuyant sur “Continu”, cela va ouvrir un menu, permettant à l’utilisateur d’obtenir les informations qu’il souhaite grâce à l’analyse de la séquence protéique.

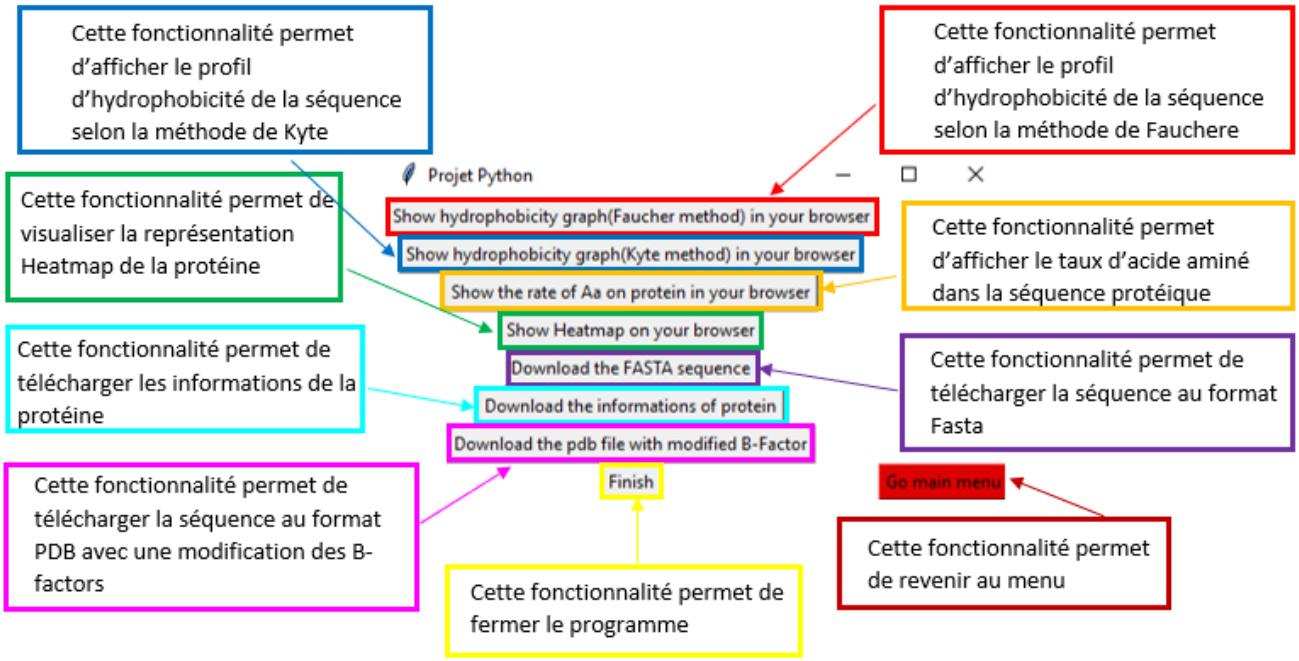


Figure 1: Menu visualisant les différentes fonctionnalités possibles à faire sur une séquence protéique

- Si on choisit de cliquer sur la deuxième proposition, il faut rentrer un code PDB valide pour pouvoir récupérer la séquence protéique sur internet. Ce code PDB doit alors être écrit par l’utilisateur sans faute, c'est-à-dire qu’il doit contenir exactement 4 caractères. Nous avons fait en sorte que l’utilisateur puisse entrer le code en majuscule ou en minuscule. L’ouverture d’un menu similaire va permettre à l’utilisateur d’obtenir les informations qu’il souhaite grâce à l’analyse de la séquence protéique.

À tout moment, il est possible de retourner à la fenêtre de choix concernant la possession ou non du fichier souhaité dans notre répertoire grâce au bouton “Go main menu”. Nous avons également pensé à gérer les erreurs grâce à l’outil “try, except, else” afin d’éviter tout bug lors de l’exécution du programme. Ainsi, si l’utilisateur entre un chemin d’accès incorrect pour sauvegarder le fichier de sortie, le pop-up suivant apparaître à l’écran : “The files will be saved in the current folder”. Cela signifie que le programme enregistrera automatiquement le fichier de sortie dans le même répertoire que celui où il se trouve. Par ailleurs, si l’utilisateur entre le nom d’un fichier PDB incorrect ou qui n’existe pas, le pop-up “You must enter a valid PDBID” apparaîtra et il faudra entrer un nouveau nom de fichier. De la même manière, lors de la sélection d’un fichier PDB présent sur le répertoire, si l’utilisateur ne sélectionne pas un fichier PDB, le pop-up “You must choose a PDB file” apparaîtra.

Lors de l’élaboration de notre programme, peu de problèmes ont été rencontrés. La première difficulté rencontrée a été la prise en main du module Biopython, car la documentation ne comporte

pas beaucoup d'exemples ce qui a augmenté notre temps de compréhension[1]. Nous avions ensuite dans l'idée de faire un Dashboard avec le module dash mais cela était trop difficile à écrire en html. Nous avons donc abandonné cette idée au vu de la difficulté de sa réalisation.

2. Récupérer le contenu du fichier

Dans un premier temps, il est nécessaire de récupérer les données contenues dans un fichier PDB. Pour cela, nous pouvons récupérer directement un fichier au format PDB stocké dans le disque dur de notre ordinateur grâce à la fonction "parcourir". Ou bien, récupérer le contenu d'un fichier au format PDB par l'interrogation du site web PDB avec le code PDB grâce à la fonction "pdb_search".

Dans le cas de la récupération d'un fichier provenant d'internet, il est nécessaire de créer un fichier transitoire à l'aide de la fonction "write_file". En effet, Biopython n'est capable de prendre que des fichiers PDB en entrée. Il faut donc créer un fichier PDB à partir des informations que l'on a pu récolter sur internet. Ce fichier transitoire sera alors supprimé automatiquement quand l'on fermera le programme avec la fonction "file_del".

Nous avons donc choisi de mettre en place la possibilité de choisir entre deux stratégies pour obtenir le contenu d'un fichier au format PDB. En passant soit par le contenu de notre disque dur, soit par des informations récoltées sur internet. Cela permet de ne pas limiter le lancement du programme, si l'utilisateur n'a pas le fichier PDB souhaité dans son disque dur ou s'il n'est pas connecté à internet par exemple. La suppression automatique du fichier transitoire permet de ne pas saturer le stockage de l'ordinateur et apparaît donc également comme un atout pour notre programme.

3. Récupération des informations importantes sur la protéine et création d'un fichier

Des informations essentielles sur la protéine sont recueillis par le programme. La fonction "exp_meth_resol" permet de récupérer la méthode de détermination expérimentale et la résolution de la protéine. En partant du chemin du fichier on obtient la méthode sous forme d'un str ainsi que la résolution sous forme d'un float dont l'unité sera l'Årmstrong.

Un fichier "1CRN_output_file.txt" va alors être créé et contenir les données suivantes : la séquence en acide aminés, la longueur de la séquence, la méthode expérimentale utilisée avec la résolution associée et la présence ou non de ponts disulfures, suivi des numéros des cystéines impliquées ainsi que la distance entre celles-ci si des ponts disulfures sont présents. La méthode pour connaître la présence ou non des ponts disulfures sera détaillée plus tard dans le rapport.

4. Construire la séquence au format fasta

La construction d'une séquence au format Fasta de cette protéine est proposée par notre programme grâce à la fonction "output_fasta". Le format Fasta est un format de fichier texte utilisé pour stocker des séquences nucléiques ou protéiques. Cette fonction va alors créer un fichier ".fasta" et ajouter sur la première ligne le signe ">" suivi du code PDB de la séquence puis le nom de la molécule suivi du nom de l'organisme ainsi que son identifiant taxonomique. Sur la deuxième ligne apparaîtra la séquence en acide aminés. L'avantage de créer un fichier au format Fasta est que l'on pourra l'utiliser directement par la suite. Par exemple, on pourra l'utiliser sur des sites internet qui demande des

fichiers dans ce format. Ce nouveau fichier créé sera automatiquement enregistré dans le même répertoire que le programme sauf s'il été précisé un autre chemin d'accès lors de la validation de la première fenêtre.

```
>1CRN|crambin|crambe hispanica subsp. abyssinica|3721
TTCCPSIVARSNFNVCRLPGTPEAICATYTGCIIIPGATCPGDYAN
```

Figure 2: Séquence 1CRN au format Fasta

5. Analyse de la composition en acide aminé de la protéine et comparaison avec la fréquence moyenne

Dans un premier temps, nous avons codé une fonction permettant de récupérer les séquences en acide aminé des fichiers sources ("get_Aa"). Cette fonction prend en entrée le chemin du fichier PDB pour le lire et renvoyer la séquence en acide aminés. Puis nous avons créé une fonction permettant d'obtenir le pourcentage d'apparition d'un acide aminé dans la protéine par rapport au taux moyen ("Aa_rate") [2]. Cette fonction prend en entrée un str représentant la séquence en acide aminés, elle comptabilise le nombre d'occurrences de chaque acide aminé puis le divise par leur nombre total. Elle renvoie un dictionnaire contenant les différents taux pour chaque acide aminé. Cela va constituer la base de notre analyse, car la composition en acide aminés permet d'obtenir de nombreuses informations sur notre séquence protéique. A l'aide de la séquence en acide aminés, nous avons créé une fonction pour obtenir le résultat précédent sous forme d'un diagramme à barres ("viz_rate"). Cela rend la compréhension du résultat beaucoup plus intuitive.

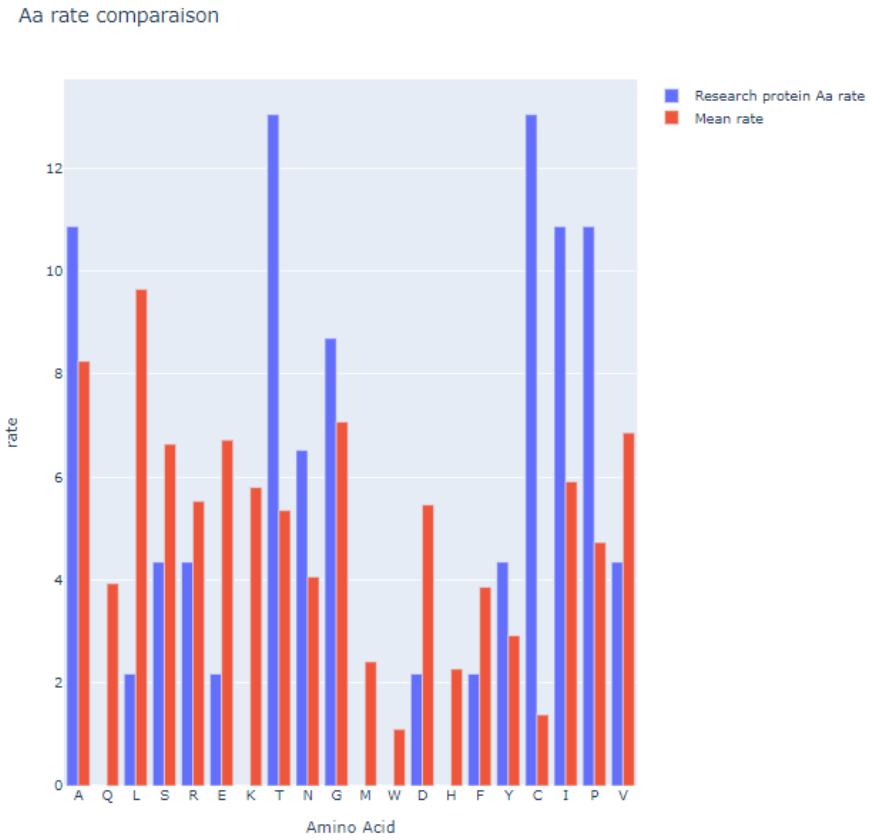


Figure 3: Graphique de comparaison du taux d'acide aminé présent dans la séquence 1CRN par rapport au taux moyen

6. Calculer le profil d'hydrophobicité

Pour calculer le profil d'hydrophobicité de la protéine nous pouvons utiliser différentes échelles d'hydrophobicité. Nous avons fait le choix de proposer deux options à l'utilisateur, l'échelle d'hydrophobicité de Fauchère ou celle de Kyte[3][4]. Pour ces deux options nous avons créé deux dictionnaires distincts correspondant aux valeurs d'hydrophobicité pour chaque acide aminé selon l'échelle. Puis la fonction "hydrophobicity_calc" permet de calculer l'hydrophobicité totale de la protéine. Pour cela, la fonction prend en entrée la séquence d'acide aminé et nous pouvons préciser la taille de la fenêtre et l'échelle d'hydrophobicité, par défaut ces deux derniers arguments sont respectivement de 9 et l'échelle de Fauchère.

On parcourt les acides aminés de la séquence un par un puis à partir de chaque acide aminé on regarde les acides aminés suivants en fonction de la fenêtre. On ajoute à un compteur, initialisé à 0 au préalable, les valeurs des acides aminés de la fenêtre. Puis on ajoute à une liste l'hydrophobicité moyenne. Et de cette façon on parcourt l'ensemble de la chaîne. Nous avons également rajouté une ligne permettant d'éviter l'erreur "out of range" en vérifiant que la fenêtre ne dépasse pas la longueur de la séquence.

Avec l'aide de plotly nous avons réalisé le graphique représentant le profil d'hydrophobicité. Le fichier de sortie est une page internet où il est possible de "jouer" avec le graphique. En effet, il est possible par exemple de télécharger le graphique, zoomer, ou encore de se déplacer.

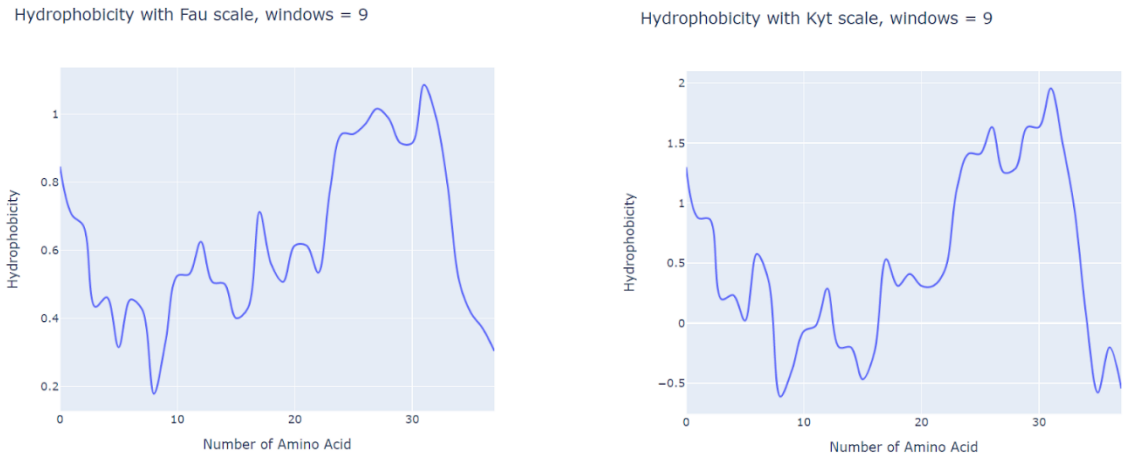


Figure 4: Profil d'hydrophobicité de la protéine 1CRN avec l'échelle de Fauchère (gauche) et l'échelle de Kyte (gauche)

7. Détecter la présence éventuelle de ponts disulfures et affichage

Notre programme permet de déterminer les cystéines qui forment un pont disulfure, lister les numéros des cystéines pontés ensembles mais également lister les cystéines non-pontés ou tenir compte de structure protéique n'ayant pas de cystéine. Un pont disulfure (S-S) est une liaison covalente forte qui, par oxydation, réunit les fonctions thiol de deux cystéines dans une protéine. Ces liaisons sont nécessaires à la stabilisation de la structure de certaines protéines (verrouillent la structure), elles peuvent aussi maintenir la liaison entre différentes chaînes ou sous-unités.

La fonction "disulfide_bridge" permet alors d'obtenir les ponts disulfures contenus dans la séquence, à partir du chemin d'accès au fichier on récupère une liste de liste qui contient la première cystéine, la deuxième cystéine puis la distance entre les deux cystéines.

```
Disulfide bound determination
bond between CYS 2 and CYS 39, distance : 2.00
bond between CYS 3 and CYS 31, distance : 2.03
bond between CYS 15 and CYS 25, distance : 2.05
```

Figure 5: Détermination des ponts disulfures dans la séquence 1CRN

8. Construire un fichier PDB de cette structure en changeant le champ « B-factor » par des valeurs tenant compte de la physico-chimie des résidus

Dans le but d'observer les acides aminés selon leurs critères physico-chimiques nous avons modifié leurs valeurs de B-factor. Sachant que les valeurs de B-factor sont comprises entre 0.00 et 999.99 nous avons créé 4 sous-familles physico-chimiques et pour chaque sous-famille nous avons attribué une valeur. On retrouve :

- Apolaire : 0.00 (ALA, LEU, GLY, MET, PHE, CYS, ILE, PRO, VAL)
- Polaire chargé négativement : 500 (GLU, ASP)
- Polaire non chargé : 750 (GLN, SER, THR, ASN, TRP, TYR)
- Polaire chargé positivement : 999 (ARG, LYS, HIS)

Nous avons créé un dictionnaire "regroup" pour attribuer à chaque acide aminé sa sous-famille. Puis dans la fonction "set_Bfact" on parcourt le fichier pdb ligne par ligne et quand on rencontre une ligne commençant par "ATOM" nous récupérons l'acide aminé. Puis, à l'aide du dictionnaire, nous attribuons à chaque acide aminé la valeur correspondant à sa sous-famille. Dans un dernier temps, nous écrivons ces nouvelles lignes dans un nouveau fichier afin de ne pas modifier le fichier initial.

Grâce à cette fonction, nous pouvons maintenant visualiser les différents acides aminés en fonction de leurs caractéristiques physicochimiques à l'aide de Pymol, en les colorant par B-factor. Les petites valeurs sont de couleur bleu et les plus grandes de couleur rouge.

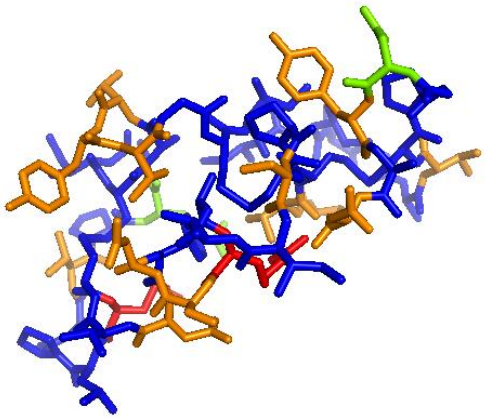


Figure 6: Visualisation à l'aide de Pymol de la protéine 1CRN en fonction des caractères physico-chimiques des résidus (bleu : apolaire, vert : polaire chargé négativement, orange : polaire non chargé, rouge : polaire chargé positivement)

9. Matrice de contact de la protéine

Afin d'observer des structures secondaires ou des repliements particuliers de la protéine, nous avons réalisé un graphique à l'aide d'une matrice de contact. Pour réaliser la matrice de contact il faut calculer la distance entre tous les carbones alpha de la protéine.

Pour cela nous avons récupéré la structure de la protéine puis on regarde les résidus. Également nous réalisons une matrice de zéro avec pour dimension (longueur et largeur) la longueur de la chaine protéique. On récupère les coordonnées des résidus et dans la matrice on ajoute la distance euclidienne entre deux résidus.

Nous avons également rajouté une condition dans cette fonction pour vérifier qu'on récupère bien les coordonnées d'acides aminés et non pas de molécules tierces tels que l'IPTG.

Pour cette fonction nous faisons appel au module numpy qui permet d'effectuer des calculs sur les matrices notamment.

Puis à l'aide de plotly nous avons réalisé le graphique représentant cette matrice de contact. Ce graphique permet de visualiser les acides aminés qui sont éloignés au niveau de la séquence mais qui sont proches spatialement. Dans notre graphique, plus la zone du graphique est foncée plus les acides aminés sont proches. Comme pour les autres fichiers de sortie qui sont des pages internet il est possible de "jouer" avec le graphique. Il est également possible d'afficher les coordonnées du graphique en positionnant la souris sur celui-ci.

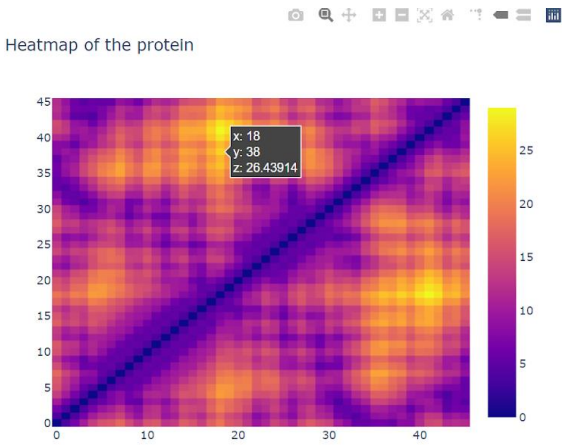


Figure 7: Heatmap de la protéine 1CRN à partir de sa matrice de contact

10. Interface graphique

L'interface graphique est le premier atout que nous avons décidé d'ajouter à notre programme. Nous avons effectué cette interface grâce au module Tkinter. Il permet de créer des fenêtres qui défilent, avec différents composants, interagissant avec l'utilisateur. Cela permet à l'utilisateur d'interagir et de choisir à souhait les informations à visualiser et dans quel ordre, comme nous avons pu le détailler précédemment. Cela rend le programme beaucoup plus intuitif pour que tout le monde puisse l'utiliser. Nous avons choisi d'utiliser une interface dynamique qui évolue en fonction des choix de l'utilisateur à l'aide de boutons ainsi que des champs de texte à remplir, tout en incluant un retour au menu en cas de besoin. Nous avions dans l'idée de faire un bouton retour en arrière, mais nous avons mis de côté cette idée car nous trouvions cela long à coder pour une utilité restreinte. Nous avons donc opté pour un retour au menu qui nous paraissait plus simple.

De plus, grâce au module “plotly”, nous pouvons visualiser les résultats à l’aide de différents graphiques adaptés à chacun, ce qui facilite fortement la visualisation et la compréhension de ceux-ci.

IV. CONCLUSION

Notre programme permet, à partir seulement du code PDB d’une protéine, de récupérer de nombreuses informations sur celle-ci. Limitant ainsi le nombre de sites à consulter pour l’utilisateur, par exemple il n’a plus besoin de consulter à la fois PDB et Expasy pour obtenir des informations sur l’hydrophobicité de la protéine, il s’agit donc d’un gain de temps. Afin d’aider l’utilisateur à exploiter notre programme nous avons fait le choix d’utiliser un menu interactif afin de rendre ce programme plus ludique et attrayant. Pour faciliter son utilisation, nous avons mis en place un système de pop-up d’erreurs permettant à l’utilisateur d’être informé de son erreur et ainsi lui donner la possibilité de corriger cela.

Ainsi, nous avons mis au point, pour faciliter d’autant plus l’utilisation du programme, un fichier ReadMe afin d’expliquer toutes les conditions nécessaires à l'utilisation de notre programme.

Afin d’améliorer notre programme nous pourrions apporter des modifications afin de gérer la condition où la protéine présente plusieurs chaînes. On pourrait également travailler sur les structures secondaires des protéines en représentant la topologie de la molécule à l’aide du module matplotlib ou nglview. Enfin, il serait intéressant de travailler avec l’outil “pipeline” dans le but d’améliorer notre programme. Il s’agit d’un agencement de logiciel/traitements lancé les uns à la suite des autres, les données ainsi obtenues en sortie d’un programme sont utilisées comme données dans le programme suivant. En d’autres termes, cela permet de regrouper plusieurs outils dans un seul programme afin d’éviter de faire tout à la main, logiciel par logiciel. Par exemple, les informations sur l’hydrophobicité de la protéine pourront être regroupées dans un seul programme, il ne faudra plus passer par PDB puis par l’outil Expasy indépendamment.

REFERENCES

[1] https://biopython.org/. [En ligne] 3 June 2021.
[2] https://web.expasy.org/docs/relnotes/relstat.html. *"AMINO ACID COMPOSITION".* [En ligne]
[3]https://resources.qiagenbioinformatics.com/manuals/clcgenomicsworkbench/650/Hydrophobicity_scales.html. *Kyte-Doolittle scale.* [En ligne]
[4] https://web.expasy.org/protscale/pscale/Hphob.Fauchere.html. *Fauchere scale.* [En ligne]