Sorbonne Université , M2 PSF, 2024 dirk.stratmann@upmc.fr

TP protein-protein docking

1 Dossier « docking »

1. Dans la machine virtuelle vous devriez avoir un dossier « docking » dans votre home-directory. Ici ce trouve des logiciels de docking, des outils et des scriptes sous python, ainsi que des fichiers d'aide et de la biblio.

2 Logiciels à utiliser

2.1 Terminal

Commandes de base en Linux / Terminal :

1. Afficher le chemin actuel : **pwd** (pour « print working directory »)
2. Changer de répertoire :
   cd relativePath/
   cd /absolutePath/
   cd .. (remonte le path)
   cd ~ (home-directory)
3. Lancer un programme qui est dans le répertoire actuel :
   ./programme
4. Compresser un dossier :
   tar cvzf archive.tar.gz dossier/
5. Décompresser un archive dans le répertoire actuel :
   tar xvzf archive.tar.gz
6. Copier :
   cp source cible

2.2 pymol

pymol se lance à partir d'un Terminal. Il s'ouvrent deux fenêtres, nous utilisons principalement la grande fenêtre noir. En bas de cette fenêtre vous pouvez taper les commandes pymol. Pour obtenir de l'aide sur une commande taper **help <commande>**.

Dans les salles de TP pymol se met par défaut dans le répertoire ~/pymol. Copier tout vos fichiers pour pymol dans ce dossier ou alors changer de path dans pymol avec la commande **cd** depuis l'invite de commande de pymol. Les commandes du Terminal comme **cd, ls, pwd** sont également disponible sous pymol, il faut juste penser d'appuyer sur la touche Echap pour voir le résultat de la commande dans la fenêtre pymol.

Sinon vous avez aussi le wiki de pymol avec beaucoup d'exemples :

http://www.pymolwiki.org/index.php/Main_Page

http://www.pymolwiki.org/index.php/Practical_Pymol_for_Beginners

3 Sites internet

3.1 Pour les exercices du TP :

PDB : www.rcsb.org

PDBsum : www.ebi.ac.uk/**pdbsum**/

3.2 Outils déjà installés dans sur la machine virtuelle :

Profit :http://www.bioinf.org.uk/software/profit/index.html

Profit-Server: http://www.bioinf.org.uk/profit/

Reduce : https://github.com/rlabduke/reduce

zdock: http://zdock.umassmed.edu/

4 Systèmes à étudier

4.1 E9-IM9

Les structures 3D du complexe E9-IM9 (COLICIN E9 DNASE DOMAIN WITH ITS COGNATE IMMUNITY PROTEIN IM9) ainsi que des formes libres des partenaires de liaison E9 et IM9 ont été résolues expérimentalement, ce qui permet de vérifier les résultats du docking.

1. Aller sur PDB et PDBsum et chercher l'entrée 1EMV (code PDB).
2. Sur la PDB télécharger le fichier 1EMV.pdb (Download Files -> PDB File (Text))
3. Avec quelle méthode expérimentale était résolu la structure du complexe ?
4. En regardant sur PDB ou PDBsum, quelle chaîne correspond à quel partenaire de liaison ? Quel est le nombre de résidus de chaque chaîne ?
5. Sur l'onglet « Prot-prot » en haut on peut obtenir des informations sur les interfaces et les types de liaisons entre les deux chaînes. Télécharger la liste des interactions (« List of interactions »).
6. Sur PDBsum on peut visualiser ou accéder aux informations des deux chaînes séparément en utilisant l'onglet « Protein » en haut, puis en bas à gauche « Protein »
7. Chercher pour chaque chaîne s'il existe des structures de la forme libre (= en dehors du complexe) :
    8. Une fois une chaîne sélectionné, cliquer sur « SAS ».
    9. Dans la liste « Aligned sequences », regarder les premières entrées (pour chaîne A : 1.-5., pour chaîne B : 1.-4.). Faites un tableau avec trois propriétés (forme libre/liée, méthode expérimentale, nombre de modèles) :

Chaine A – IM9 :

Code PDB	forme libre/liée	méthode expérimentale	nombre de modèles

Chaine B – E9 :

Code PDB	forme libre/liée	méthode expérimentale	nombre de modèles

10. On a alors une entrée pour la forme libre de E9 (1FSJ:B) et deux entrées pour la forme libre de IM9 (1IMP:A, 1IMQ:A). Pour faciliter le docking on choisit IMQ:A pour IM9, pourquoi ?

11. Regarder 1FSJ avec la PDB. On voit que 1FSJ contient quatre chaînes (B,C,D,E). Regarder sur le cadre de visualisation les différents modes (« Asymmetric Unit », « Biological Assembly »). Expliquer brièvement la définition de « Asymmetric Unit » et « Biological Assembly » et leur différence (on obtient une explication en cliquant sur « ? »):
12. Pourquoi est-ce qu'il suffit de prendre juste la chaîne B de 1FSJ comme modèle pour E9 ?
13. Regarder 1IMQ avec la PDB. Pourquoi le « Asymmetric Unit » et le « Biological Assembly » montrent la même chose ?
14. Télécharger 1FSJ.pdb et 1IMQ.pdb. Copier tous les trois fichiers PDB (1EMV, 1FSJ, 1IMQ) dans votre dossier ~/docking depuis le dossier ~/Downloads.
15. Afficher les fichiers trois fichiers PDB avec un éditeur de texte (par exemple dans le Terminal taper: **cd ~/docking** , puis **kate 1FSJ.pdb &**). La structure d'un fichier PDB est divisée en deux parties : l'entête et les données structurales (toutes les lignes qui commencent avec ATOM). Le format des lignes ATOM est le suivant :
    ATOM, index, nom d'atome, type d'acide aminé, id chaine, numéro résidu, x, y, z, ...
    Plus de détail :
    http://www.wwpdb.org/documentation/file-format-content/format33/sect9.html#ATOM
16. 1IMQ contient des types d'atomes que 1FSJ et 1EMV ne contiennent pas. C'est quel type d'atome ?
17. D'où vient cette différence entre 1IMQ et 1FSJ/1EMV ?
18. Créer un fichier 1FSJ_B.pdb qui ne contient que la chaîne B (avec ou sans l'entête). Utiliser un

simple éditeur de texte comme **kate** pour enlever les autres chaînes.

19. Certains programmes de docking n'acceptent pas des structures avec des hydrogènes (=> zdock) , pour d'autres c'est l'inverse. Utiliser le programme « reduce » pour créer des fichiers PDB sans et avec hydrogènes :
    - Ouvrir un Terminal dans le répertoire ~/docking
    - Taper « reduce » pour obtenir une aide
    - Taper :
    ./**reduce -Trim 1IMQ.pdb > 1IMQ_sansH.pdb**
    - Taper :
    ./**reduce 1FSJ_B.pdb > 1FSJ_B_avecH.pdb**

5 zdock

5.1 Lancer zdock

1. Pour voir les options de zdock, lancer zdock depuis un Terminal dans votre dossier de travail:
   /home/user/docking/software/docking/zdock/x64/zdock
2. L'utilitaire « mark_sur » de zdock permet de préparer les deux fichiers PDB de E9 et Im9 :
   /home/user/docking/software/docking/zdock/x64/mark_sur 1FSJ_B.pdb receptor.pdb
   /home/user/docking/software/docking/zdock/x64/mark_sur 1IMQ_sansH.pdb ligand.pdb
   Or cet exécutable a été compilé sur une vieille version de Linux et ne fonctionne plus sur la version actuelle. Pour cela les deux fichiers sont fournis : copier les deux fichiers receptor.pdb et ligand.pdb depuis ~/docking/data/zdockTest/ dans votre dossier de travail.
3. Regarder ici pour savoir ce qui fait mark_sur : ~/docking/aides/zdock/zdock_output_file.shtml
4. Pour vérifier ce qui fait mark_sur, comparer le fichier PDB que génère mark_sur avec le fichier PDB à l'entrée. Utiliser aussi un programme de visualisation comme pymol. Exemple avec pymol : Depuis pymol en étant dans le bon dossier (vérfier avec **pwd**) :
   load 1FSJ_B.pdb
   Dans pymol taper les commandes (voir en haut pour l'aide sur pymol) :
   hide everything
   show dots
   spectrum b
   Faites la même chose depuis un deuxième Terminal avec **receptor.pdb**
   A quoi correspondent les deux colorations ?
5. Depuis votre dossier de travail lancer zdock 3.0.2 avec les options par défaut :
   /home/user/docking/software/docking/zdock/x64/zdock -R receptor.pdb -L ligand.pdb -o zdock.out
   Un run peut durer 5-10 minutes, ouvrir alors un autre Terminal pour continuer sur autre chose. Vous pouvez lancer un deuxième run de zdock en parallèle (voir plus bas).
6. Regarder le fichier zdock.out par rapport sa documentation :
   ~/docking/aides/zdock/zdock_output_file.shtml
7. Relancer zdock en initialisant le générateur de nombres aléatoires avec une valeur quelconque (« seed » en anglais). Faites attention de changer le nom du fichier de sortie :
   /home/user/docking/software/docking/zdock/x64/zdock -S 13128 -R receptor.pdb -L

ligand.pdb -o zdock2.out
Noter la valeur de seed utilisée (à la place de 13128) :

8. Quand un run de zdock a terminé, copier **create.pl**, **create_lig**, **receptor.pdb, ligand.pdb, 1EMV.pdb** et **zdock.out** dans un nouveau dossier. Générer en suite dans ce dossier les 10 meilleurs modèles avec **create.pl** (taper **/home/user/docking/software/docking/zdock/x64/create.pl** pour voir les options).

1. Répéter la même chose pour zdock2.out dans un autre dossier.
2. Pourquoi/Comment est-ce que create.pl peut générer des modèles 3D tout atome du complexe en utilisant uniquement les structures libres et zdock.out ?

5.2 Analyse des résultats

Si besoin, Zdock a déjà été exécuté deux fois sur les structures de départ 1FSJ et 1IMQ. Les résultats se trouvent dans le dossier TPpymol/docking/run1 et TPpymol/docking/run2 une fois l'archive TPpymol.tar.gz décompressé :

Utiliser « Krusader » ou le gestionnaire de fichier standard de Linux pour décompresser l'archive TPpymol.tar.gz dans votre home-directory. TPpymol.tar.gz doit être téléchargé depuis le site de l'UE.

Ouvrir un Terminal puis aller dans le dossier crée, normalement **cd TPpymol**

Puis aller dans le sous-dossier **docking**

5.3 Pymol

1. Lancer pymol, puis aller dans un des deux dossiers (run1 ou run2) avec les 10 meilleurs modèles avec la commande **cd** dans pymol.
2. On utilise un scripte python « loadZdock.py » pour charger toutes les modèles générés en tant qu'un seul objet « mov » dans pymol. Regarder ici http://www.pymolwiki.org/index.php/Load sous « User Comments/Examples » pour avoir plus d'informations sur ce scripte python. Vérifier que les complexes sont chargés dans le bon ordre.
3. Pour utiliser loadZdock.py dans pymol, il suffit de taper dans pymol:
   run ~/TPpymol/docking/loadZdock.py
   voir ici pour plus de détail: http://www.pymolwiki.org/index.php/Running_Scripts
4. Colorer les deux chaînes en deux couleurs différentes et utiliser le mode cartoon :
   hide everything
   show cartoon
   color red, chain A
   color blue, chain B
5. Vous pouvez défiler parmi les différents modèles en utilisant la barre de type lecteur de musique en bas à droite dans pymol.
6. Charger maintenant aussi la structure de référence 1EMV avec la commande **load** de pymol.
7. On cherche à superposer la partie E9 de 1EMV avec la partie E9 des modèles (objet « mov » dans pymol). Pour cela il faut créer des sélections de ces parties (help select) :
   select ref, 1EMV and chain B
   select mobile, mov and chain B

8. En suite on peut utiliser la commande **fit** (help fit) pour superposer les deux sélections.
9. Pour rendre la chose plus jolie mettez aussi 1EMV en mode cartoon :
   hide everything, 1EMV
   show cartoon, 1EMV
10. Qu'est-ce qu'on peut dire sur les 10 meilleurs modèles obtenu avec zdock ?
11. Répéter cette analyse avec les résultats du deuxième run de zdock (zdock2.out). Pour cela mettez les commandes pymol dans un fichier texte (ex : zdock.pymol), puis le lancer avec la commande pymol : **@zdock.pymol**

    Copier votre scripte zdock.pymol ici :

12. Comparer les deux résultats :

5.4 CAPRI

1. Pour obtenir une analyse plus quantitative de l'accord des modèles avec la référence, on utilise les critères de CAPRI. Quatre mesures quantitatives ont été établie pour CAPRI : fraction of native contacts (fnat), fraction of non-native contacts (fnonnat), ligand-RMSD et interface-RMSD. Lisez dans l'article correspondant la partie qui explique comment sont calculées ces quatre grandeurs (voir parties jaunes dans ~/docking/biblio/2003_MendezWodak_Proteins_CAPRI.pdf). Expliquer ici vous-même comment sont calculées ces quatre grandeurs et ce que mesure chacune :

5.5 CAPRI.py

1. Le programme CAPRI.py utilise la librairie Biopython (~/docking/aides/biopython) pour calculer les quatre paramètres : fraction of native contacts (fnat), fraction of non-native contacts (fnonnat), ligand-RMSD et interface-RMSD. Copier CAPRI.py depuis ~/docking/software/scripts/CAPRI.py dans les deux dossiers correspondants au zdock.out et zdock2.out.
2. Dans un Terminal depuis un de ces deux dossiers lancer CAPRI.py sans paramètres pour obtenir la liste des paramètres :

   python CAPRI.py
3. Lancer CAPRI.py avec les bons paramètres en faisant attention quelle chaîne est le récepteur et quelle chaîne est le ligand. Recopier ici la ligne de commande utilisée :

4. Si tout c'est bien passé, on obtient un fichier CAPRI_results.out.
5. En analysant CAPRI_results.out qu'est-ce qu'on peut dire par rapport l'analyse qualitative avec pymol ?
6. Est-ce que obteniez des solutions avec des étoiles d'après l'évaluation de CAPRI (voir 2003_MendezWodak_Proteins_CAPRI.pdf) ? Recopier ici CAPRI_results.out (mis en forme

n'est pas nécessaire) et noter les étoiles s'il y en a :

7. Lancer CAPRI.py depuis le dossier avec zdock2.out et recopier ici CAPRI_results.out (mis en forme n'est pas nécessaire) et noter les étoiles s'il y en a :

8. Conclusion sur les deux runs avec zdock sur E9-IM9 :

5.6 Profit

Profit est un programme qui permet de calculer des valeurs RMSD (interface-RMSD, ligand-RMSD, ...) entre deux protéines après les avoir superposées au mieux.

1. Pour calculer le ligand-RMSD sur les trois premiers modèles on utilisera le programme **profit** à partir d'un Terminal. Lancer profit depuis le dossier qui contient les dix meilleurs modèles de zdock.out avec **profit** Vous pouvez obtenir de l'aide sur chaque commande de profit en utilisant la commande **help <commande>**. Sinon vous avez aussi la documentation complète dans ~/docking/aides/profit.
2. Charger la structure de référence 1EMV.pdb avec :
   refe 1EMV.pdb
3. Puis un des trois premiers modèles :
   mobi complex.1.pdb
4. Définir les types d'atomes sur lesquelles on calcule le RMSD, ici les atomes lourds du squelette pétitdique :
   atoms CA,C,N,O
5. La commande **zone** permet de définir les résidus qui doivent être utilisés pour accorder au mieux la structure mobile (complex.1.pdb) sur la structure de référence (1EMV.pdb). Pour le calcul du ligand RMSD cela doit être la partie du récepteur (= le plus grand des deux partenaires de liaison), alors ici la chaîne B. On pourrait alors mettre comme paramètre à **zone** :
   zone B*
6. Ensuite la commande fit accorde au mieux la structure mobile (complex.1.pdb) sur la structure de référence (1EMV.pdb) en utilisant les paramètres donné avec **atoms** et **zone** :
   fit
   Ceci va donner un message d'erreur, car la nombre et/ou la numérotation des résidus n'est pas la même pour 1EMV.pdb et complex.1.pdb. Corriger cela en effaçant d'abord la zone définie :
   zone clear
   Puis en spécifiant l'intervalle de résidus communs (regarder les fichiers pdb) :
   zone B#-###:B#-###
   (# = chiffre, avant le « : » => référence, après le « : » => mobile).
   Mettre ici la commande utilisé :

Relancer **fit**, si tout va bien vous aurez le RMSD de la superposition optimale entre le modèle et la référence de la partie récepteur.

7. Une fois la partie récepteur bien superposé, on peut calculer l'accord de la partie ligand (=ligand RMSD). Pour cela, il faut définir une nouvelle zone sur laquelle doit être calculé le RMSD sans bouger les structures avec la commande **rzone** :
   rzone A*
   Ceci donne encore un message d'erreur, comme pour **zone B*** en haut. En examinant les fichiers PDB, ajuster le paramètre de rzone (après un **rzone clear**) et recopier la bonne commande ici :

8. Noter la valeur RMS obtenue, c'est le ligand RMSD pour le premier modèle. Répéter la même chose pour les trois premiers modèles. Sans quitter profit, il suffit de réutiliser les commandes suivantes :
   mobi complex.X.pdb
   fit
   rzone ...
   Noter ici les trois valeurs obtenues :
9. Comparer les valeurs du ligand-RMSD (l-RMSD) de CAPRI_results.out avec les trois valeurs obtenues par profit, normalement elles doivent être similaires. Est-ce que c'est le cas (oui/non) ?

5.6.1 Différences entre structures *bound* et *unbound*

1. Utiliser profit pour calculer le RMSD entre les formes liées (*bound*) et libres (*unbound*) de E9-IM9. Copier les commandes profit utilisées ici :
2. Noter et commenter les valeurs RMSD obtenues pour E9 et IM9 ici :

6 Obtenir de meilleur résultats

Dans cette partie nous allons simplifier le problème du docking soit en partant des structures issues du complexe de référence (« bound-bound » docking) soit en restreignant la recherche autour du site d'interaction.

6.1 Bound-bound docking

On commence à docker les formes liées (bound-bound) du complexe.

1. Expliquer pourquoi ceci est un cas plus simple par rapport un docking avec des formes non-liées (unbound-unbound) ?
2. Lequel des deux types de docking retrouve-t-on quand on ne connaît pas la structure du complexe ?
3. Créer un nouveau dossier « 1EMV_bound ». Préparer là-dedans un fichier receptor.pdb et un

fichier ligand.pdb à partir du fichier 1EMV.pdb contenant uniquement les lignes ATOM de la chaine correspondante (récepteur = la plus grande des deux chaines).

4. Utiliser zdock (voir au début du TP) pour docker ces deux structures « bound ».
5. Évaluer les 10 meilleurs modèles avec CAPRI.py. Copier-coller ici CAPRI_results.out en indiquant les étoiles d'après l'évaluation de CAPRI :

6. Conclusion ?

6.2 Restreindre la recherche autour du site d'interaction

1. Créer un nouveau dossier « 1EMV_interface » et copier y 1EMV.pdb et le script getNonInterface.py (même dossier que CAPRI.py).
2. Ouvrir 1EMV.pdb avec un éditeur de texte et enlever toutes les lignes qui ne commencent pas avec ATOM.
3. Lancer le script getNonInterface.py (attention aux options!) sur 1EMV.pdb pour obtenir deux listes avec les résidus qui ne sont pas à l'interface (getNonInterface_receptor.out et getNonInterface_ligand.out). Lancer aussi le scripte pymol généré :
   pymol 1EMV.pymol
4. Copier les structures « unbound » **receptor.pdb** et **ligand.pdb** depuis le dossier de la première partie du TP.
5. Lancer **block.pl** sur les deux:
   /home/user/docking/software/docking/zdock/x64//block.pl receptor.pdb getNonInterface_receptor.out > receptor_blocked.pdb

   /home/user/docking/software/docking/zdock/x64/block.pl ligand.pdb getNonInterface_ligand.out > ligand_blocked.pdb
7. Lancer zdock avec receptor_blocked.pdb et ligand_blocked.pdb.
8. A quoi sert **block.pl** ?
9. Pourquoi est-ce qu'on veut « bloquer » certains résidus ?
10. Si on ne connaît pas la structure du complexe, comment est-ce qu'on peut obtenir des informations sur les résidus à bloquer ?
11. Analysez les résultats du docking sur les dix meilleurs modèles avec pymol. Décrivez ici vos résultats :

12. Analysez les résultats du docking sur les dix meilleurs modèles avec CAPRI.py. Copier-coller ici CAPRI_results.out en indiquant les étoiles d'après l'évaluation de CAPRI :