Structural and Data-Driven Approaches to Drug Discovery

Fergus Boyles and Tom Hadfield, Doctoral Training Centre, University of Oxford, 30NOV2021

Protein-Ligand Docking

In this practical we will explore how protein-ligand docking to rapidly screen compound libraries against a protein target. Our target for today is the cAMP-dependent protein kinase (PDB: 1Q8T)

Docking using GOLD

Lots of docking engines exist but today we'll be using GOLD, maintained by the Cambridge Crystallographic Data Centre. Some docking scripts, as well as the data needed for this exercise are available at: https://github.com/tomhadfield95/DTC_virtual_screening. The protein and ligands have already been prepared for you to dock straightaway, so for now you don't need to do any painful preprocessing!

Requirements

Download the github repo containing the required scripts/files, either by running `git clone https://github.com/tomhadfield95/DTC_virtual_screening` in the terminal, or by downloading it from the github web interface. If you don't have access to a terminal, or haven't used one before, please alert one of the demonstrators as soon as possible - we have an alternative practical for you to get on with!

You'll need to activate the `gold` conda environment by running `conda activate gold` in the terminal. Test that you'll be able to run the docking scripts by running `python dock_gold.py --help` in the terminal (after cd'ing into the DTC_GOLD_docking directory)- if this returns some kind of error, please let one of the demonstrators know. If it runs smoothly then you should be good to continue with the rest of the practical.

Part 1: Docking the Crystal Ligand

The structure of cAMP-dependent protein kinase deposited in the PDB (ID 1Q8T) was in complex with a ligand. We therefore know with a high degree of certainty what the true binding pose of the ligand looks like; if GOLD predicts that the ligand would adopt a radically different pose then that would be a cause for concern!

Dock the ligand contained in the file `1q8t_ligand.sdf` into the protein. You can do this using the python script `dock_gold.py` which can be run with the following parameters:

```
     python dock_gold.py <ligand sdf file> <protein pdb file> \
                         -o <output sdf file> -c <crystal ligand>
```

For more information about what each of the different parameters does, simply run: `python dock_gold.py --help`

GOLD will generate a number of poses for the ligand (default = 10) and score them using its scoring function. The top ranked pose should be saved wherever you specified `<output sdf file>,` and you should be able to view it in PyMol.

Inspect the docked ligand. Does it adopt a similar conformation to the crystal ligand or not? Calculate the RMSD between the two conformations. If the two conformations are very different, try docking the ligand again, this time using the `--all` flag in `dock_gold.py.` This will save all of the poses generated by GOLD, not just the top-ranked one - are any of the other conformations similar to the crystal pose?

Part 2: Virtual Screening

One strategy for discovering new drugs is to simply test lots and lots of compounds against a target and see if any of them bind with high affinity. This approach, called High Throughput Screening, is very expensive if we experimentally test each compound in a very large library, so one way of reducing the number of compounds we have to test in the lab is to dock each compound, predict computationally which ones will bind and test the most promising ones experimentally.

The file `vs_library.sdf` contains 1000 ligands. ***Dock them against the c-AMP dependent protein kinase.*** You can reduce the time required to dock the ligands by using the `-mp` flag, which tells GOLD to use multiprocessing.

Once you've successfully docked all the ligands in this mini virtual screening library, ***rank them based on their docking score***. You can read sets of molecules into python using the `RDKit` function `rdkit.Chem.SDMolSupplier(<path to ligands>)`. Once you've loaded the ligands into RDKit, you can access the saved properties assigned to a molecule called `mol` by running `mol.GetProp(<property name>)`.

What rank was attained by the 'ground truth' ligand? Would we have identified it as a potential binder if we didn't already know it bound to the protein?

Part 3: Iterative Design & Limitations

Now that we've ranked each of the ligands in our library, inspect some of the top-ranked ones in PyMol. Are there any common features, such as interactions with a specific residue? How might we use these results as a basis for designing additional compounds for the next round of a drug discovery campaign?

In addition, what are the limitations of our experiment? Did we use any information in this experiment that we wouldn't have had access to in a real world virtual screening campaign? How might that affect our chances of success?