Structure-Based Virtual Screening
Docking Experiment

- Download Protein 5L58 https://www.rcsb.org/structure/5L58
    - Delete Liegands
        - “remove resn 6MX”
        - “remove resn NDP
        - “save 5L58_clean.pdp

- Download Liegand 6MX separately https://www.rcsb.org/ligand/6MX

- Move both files to wherever your PyMOL is saving files in my case
    - Open a terminal to location of PyMOL saves and write
        - “mv ~/Downloads/6MX_ideal.sdp ~/”
        - “mv ~/Downloads/5L58.pdb~/”

- Change files to .pdbqt types
    - In terminal use command
        - mk_prepare_receptor.py -i 5L58_clean.pdb -o 5L58 -p -a
        - mk_prepare_ligand.py -i 6MX_ideal.sdf -o 6MX.pdbqt

- Get location of docking coordinates
    - Open 5L58.pdb
        - “select lig, resn 6MX’
        - “center lig”
        - “get_position”
            - It should show cmd.get_position: [14.393, 22.061, 182.532]

- Create a config.txt
    - “nano config.txt’ in the terminal and paste
        - receptor = 5L58.pdbqt
          ligand = 6MX.pdbqt

          center_x = 14.393
          center_y = 22.061
          center_z = 182.532

          size_x = 22
          size_y = 22
          size_z = 22

          exhaustiveness = 16
          num_modes = 10
        - control + o = write
        - control + x = exit

- Run the docking
    - “./vina --config config.txt --out docked_6MX.pdbqt”

- The results that output in the terminal
- Results show

```
Computing Vina grid ... done.
Performing docking (random seed: -1243672540) ...
0%   10   20   30   40   50   60   70   80   90   100%
|----|----|----|----|----|----|----|----|----|----|
**************************************************

mode |   affinity | dist from best mode
     | (kcal/mol) | rmsd l.b.| rmsd u.b.
-----+------------+----------+----------
   1       -8.985          0          0
   2       -8.847      2.413      3.255
   3       -8.843      1.794      4.304
   4       -8.738      2.279      10.29
   5       -8.432      2.213      4.575
   6       -8.352      3.301      9.536
   7       -8.326      2.099      4.528
   8       -8.269      2.259      9.367
   9        -8.18      3.418      4.715
   10       -8.164      1.673      9.182
```

- Rule of thumb:

Affinity	**Interpretation**
-6	Weak
-7	Moderate
-8	Good
-9	Strong

Mode	**RMSD**	**RMSD**	**Meaning**
1	0 (reference)	<2 Å	very similar pose
2	2.41 Å	2–3 Å	somewhat similar
3	1.79 Å	>5 Å	different orientation

Interpretation:
Your results show:
Your best score: -8.985 kcal/mol That is **very good** for a small molecule inhibitor.
And it makes sense because **6MX is the actual ligand in the crystal structure**, so docking rediscovered the real binding pose.

several poses within **~2 Å** meaning the docking is **stable and consistent**
That's a good sign.
"The molecule 6MX likely binds strongly to this cancer protein, and the docking algorithm rediscovered the binding pose."

Test Multiple Molecules

- Download a subset of molecules below is an example of 500,000 .sdf files from the PubChem Database but there are many holding millions of these molecule files such as Zinc or ChEMBL
- The following command gets 500,000 files from PubChem
    - “curl -L https://ftp.ncbi.nlm.nih.gov/pubchem/Compound/CURRENT-Full/SDF/00000001_00025000.sdf.gz -o molecules.sdf.gz”
- The next unzips the folder
    - “gunzip pubchem.sdf.gz”
- The next counts the number of files to show there should be around 500,000
    - “grep '$$$$' pubchem.sdf | wc -l”
- Select number of files you want running python script
    - “python select_no_files.py”
- Count number of files in folder to make sure its 100
    - “grep '$$$$' first100.sdf | wc -l”
- Split molecules into separate files
    - “python split_molecules.py”
- Folder for prepared ligands
    - “mkdir -p ligands_ready_sdf”
- Prepare the ligands to sdf
    - “python convert_molecules_ready_sdf.py”
- Make folder for .pdbqt files
    - “mkdir -p ligands_pdbqt”
- Convert all sdf files to .pdbqt files **********
    - “for f in ligands_ready_sdf/*.sdf; do
        base=$(basename "$f" .sdf)
        mk_prepare_ligand.py -i "$f" -o ligands_pdbqt/$base.pdbqt
      done”
- Check the files have been converted
    - “ls ligands_pdbqt/*.pdbqt | wc -l”
- Run the bash script to test multiple molecules against the protein
    - “bash multiplemolecules.txt”
        - Or use Faster Version using all CPUs
            - “`./multiplemolecules.sh`”
- Double check it’s running open a new terminal
    - “tail -f scores.csv”
- Show the largest negative values at the top
    - “(head -n1 scores.csv && tail -n +2 scores.csv | sort -t, -k2 -g) > ranked_scores.csv
      head ranked_scores.csv”

Test Top Hit Molecules

- Look to see if the molecules have already been tested against the protein
    - “python top_hits_identified.py
    - “python already_discovered_molecules.py”
    - This will give you a list of proteins, make sure you chose the correct protein to screen molecules against use the CHEMBL ID and place it in the script under TARGET_CHEMBL_ID then run the script again
    - This will give top_hits_checked.csv
    - The most useful columns are:

        - pubchem_cid → confirms the exact compound exists in PubChem
        - chembl_id → exact ChEMBL molecule match
        - tested_on_target → whether ChEMBL already has activity data for your chosen target
        - best_activity_type, best_activity_value, best_activity_units → the strongest recorded assay row it found

        - tested_on_target = yes
          your docking likely rediscovered something already tested against that protein
        - chembl_id exists but tested_on_target = no
          that is more interesting: known compound, but possibly **not yet tested on your target**
        - no PubChem CID and no ChEMBL match
          either the structure is unusual, the exact standardization differs, or it may be a less common entry

View Protein and Ligand to see if it binds in the correct place

- Change molecule to sdf
    - “obabel docking_outputs/mol_87_out.pdbqt -O mol_87_docked.sdf”
    - In Pymol load proten .pdb file and type “**load mol_87_docked.sdf”**

        PyMOL>select native, resn 6MX
        PyMOL>show sticks, native
        PyMOL>color orange, native (Shows the inhibitor from crystal structure**.)**
        PyMOL>show sticks, mol_87_docked
        PyMOL>color yellow, mol_87_docked (show your docked hit)
        PyMOL>hide everything, 5L58
        PyMOL>show cartoon, 5L58
        PyMOL>color cyan, 5L58 (show the protein)
        PyMOL>zoom native (zoom into binding pocket)
        PyMOL>select pocket, byres (5L58 within 4 of mol_87_docked)
        PyMOL>show sticks, pocket
        PyMOL>color green, pocket (amino acids interacting with your molecule)

PyMOL>set cartoon_transparency, 0.6 (make the native ligand transparent

Redocking with stricter parameters

There are thousands of cancer proteins and billions of molecules available online you're able to test against these cancer proteins.

Enamine REAL Database has 36 billion molecules to go through, its estimated there is 10^60 drug like molecules to test against.

Because the number of possible molecules is enormous, finding one that truly inhibits a cancer protein and works as a safe drug is extremely rare. However, it is relatively easy to find molecules that computationally dock to a cancer protein, especially ones that have never been experimentally tested, although most of these will not work in real biological systems.

Before a lab would normally take interest, researchers typically add several additional computational steps.

Typical pipeline:

1. **Docking** (you have done this)
2. **Redocking validation** (you did with 6MX)
3. **Docking multiple candidate molecules**
4. **Rescoring / consensus docking**
5. **Molecular dynamics simulations**
6. **ADMET prediction**
7. **Binding free energy estimation**
8. **Experimental validation**

Only after step 8 do we know if a molecule really works.

Docking Multiple Candidates

- Script to search for multiple molecules (top 20) similar to the one you docked
    - “python get_similar_molecules.py
        - Change SEED_SDF = "ligands_sdf/mol_87.sdf" to whatever and wherever your seed molecule is
- Loop back to **********
    - Then run same docking procedure on 20 similar molecules and view and compare outputs.
- Choose the best one, if similar molecules don’t dock, then this is likely not a good molecule to begin with, just a fluke.

Rescoring / Consensus docking

- Check a different scoring method to see if you get the same results or it’s off. If it’s off it would suggest it’s not the best molecule. If it scores well with both then you may be on to something

```
./vina \
  --receptor 5L58.pdbqt \
  --ligand mol_87.pdbqt \
  --scoring vinardo \
  --center_x 14.394 \
  --center_y 22.061 \
  --center_z 182.532 \
  --size_x 20 \
  --size_y 20 \
  --size_z 20 \
  --exhaustiveness 8 \
  --cpu 1 \
  --out mol_87_vinardo_out.pdbqt
```

- AD4 scoring is done by first setting up maps
    - First set up a grid parameter file
        - mk_prepare_receptor.py \
          -i 5L58_clean.pdb \
          -o 5L58 \
          -p \
          -g \
          --box_center 14.394 22.061 182.532 \
          --box_size 20 20 20 \
          -a
    - Generate maps
        - autogrid4 -p 5L58.gpf -l 5L58.glg
    - Run docking and scoring
        - ./vina \
          --ligand mol_87.pdbqt \
          --maps 5L58 \
          --scoring ad4 \
          --exhaustiveness 8 \
          --cpu 1 \
          --out mol_87_ad4_out.pdbqt

Once you have the results of all three tests, compare with the original ligand, if results are better strong contender, if they're not discard.

Coming soon

- Molecular dynamics simulations
- ADMET prediction
- Binding free energy estimation
- Experimental validation