

Using MolAICal: 3D drug design in the pocket of GCGR by artificial intelligence and *de novo* method

Qifeng Bai

Email: molaical@yeah.net

Homepage: <https://molaical.github.io>

School of Basic Medical Sciences

Lanzhou University

Lanzhou, Gansu 730000, P. R. China

1. Introduction

In this tutorial, the standard protocol of MolAICal is introduced for the drug design of glucagon receptor (GCGR) by artificial intelligence and *de novo* method. It will help the pharmacologist, chemists and other scientists design rational drugs according to three-dimensional active pocket of proteins.

2. Materials

2.1. Software requirement

- 1) MolAICal: <https://molaical.github.io>
- 2) UCSF Chimera: <https://www.cgl.ucsf.edu/chimera/>

2.2. Protocol files

All the necessary tutorial files are downloaded from:
<https://github.com/MolAICal/tutorials/tree/master/001-AIGrow>

3. Procedure

3.1. Dealing with receptor

1. Download the human glucagon receptor (GCGR) PDB ID: 5EE7:
<http://www.rcsb.org/structure/5EE7>
2. Start UCSF Chimera: File→Open, load the PDB file 5EE7.pdb in your directory (see Figure 1).

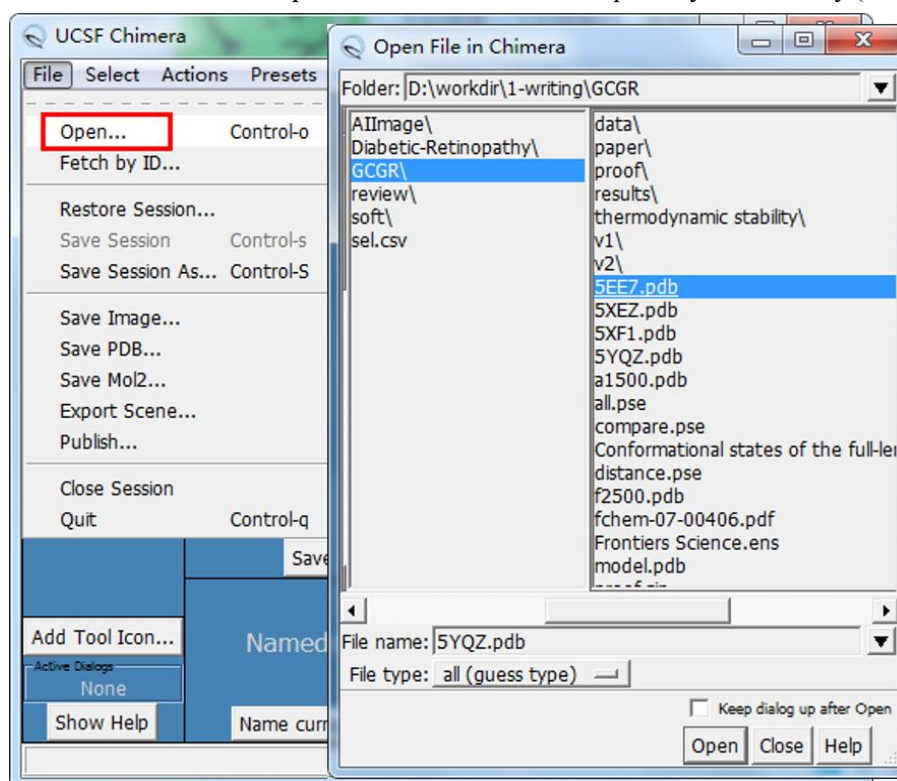


Figure 1. Loading a molecule.

3. Select and delete the chosen molecules.
Select the no useful ligand: Select→Residue→OLA (see Figure 2).

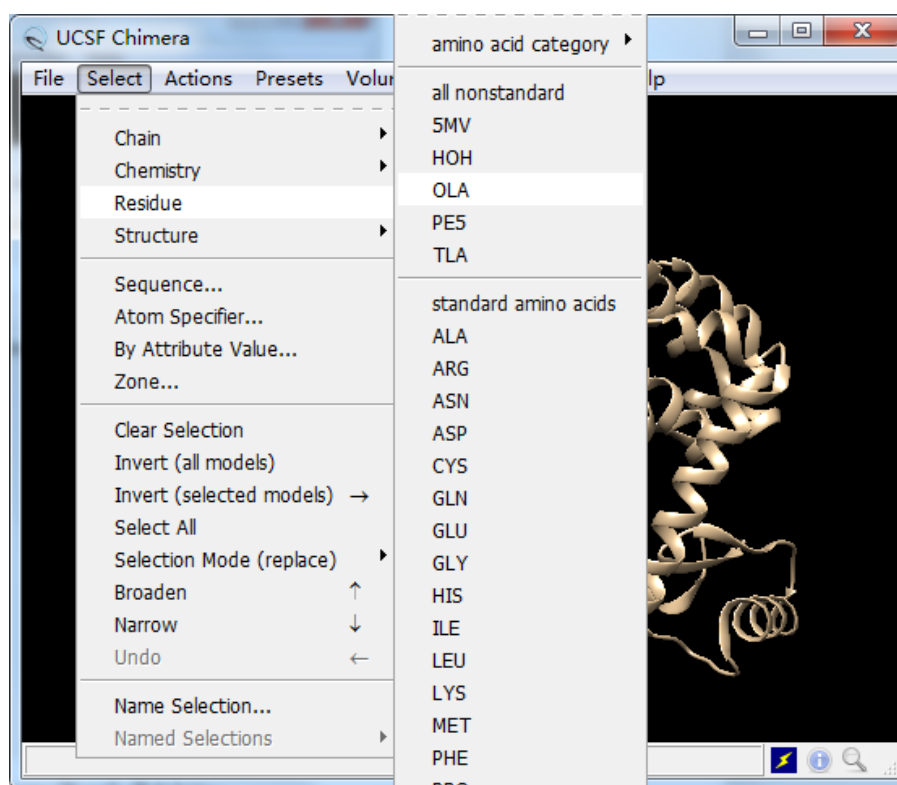


Figure 2. Selecting a molecule.

Delete no useful ligand: Actions→Atoms/Bonds→delete (see Figure 3).

We only show the simple example, you can continue to delete HOH, PE5 TLA, *etc.* in the same way.

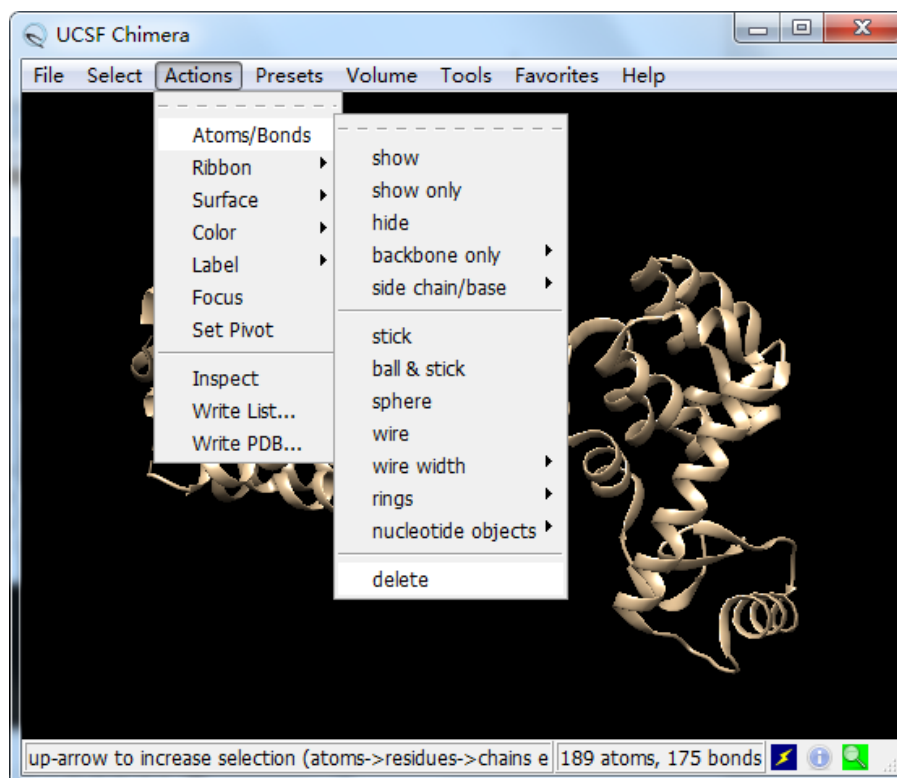


Figure 3. Deleting the selected molecule.

4. Add hydrogen. Tools→Structure Editing→AddH (see Figure 4).

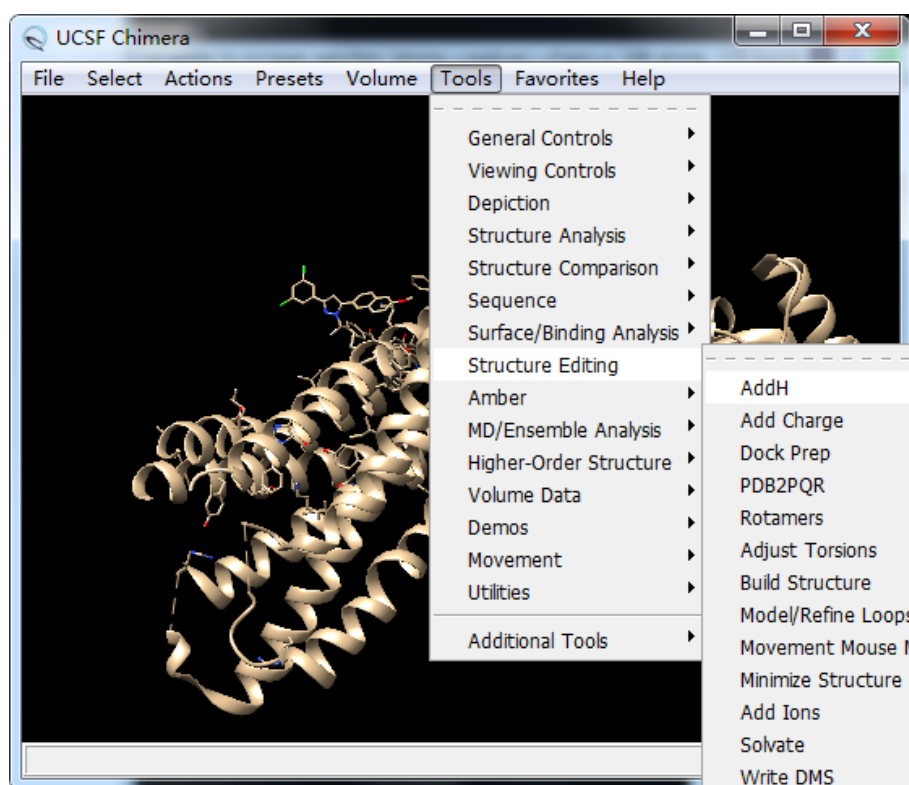


Figure 4. Adding hydrogen.

Notice: because MolAICal deals with receptors by the Amber force field, please delete hydrogen atoms before adding hydrogen by UCSF Chimera. If the receptor had the redundant hydrogen, please delete it, then added it. The detail procedures: firstly, select hydrogen: Select→Chemistry→element→H, then delete selection: Actions→Atoms/Bonds→delete (see Figure 3). Here, the crystal GCGR has no redundant hydrogen atoms, so this step is omitted.

5. Save the coordinate: GCGRH.pdb. Saving GCGRH.pdb can be as a checkpoint in the process of drug design. You can start this step to deal with GCGR directly (see Figure 5).

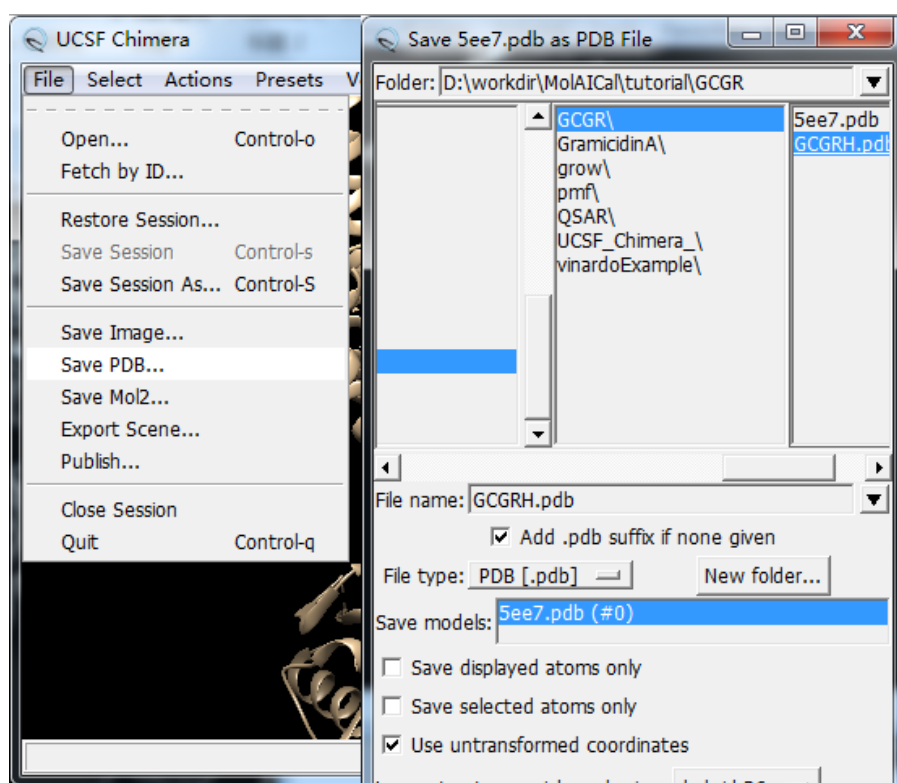


Figure 5. Saving the coordinates of receptor and ligand as PDB file.

6. Selecting ligand 5MV and deleting it (see Figure 6 and 7). Save the coordinates of apo-state GCGR (no ligand bound): GCGRNoLigand.pdb (see Figure 8).

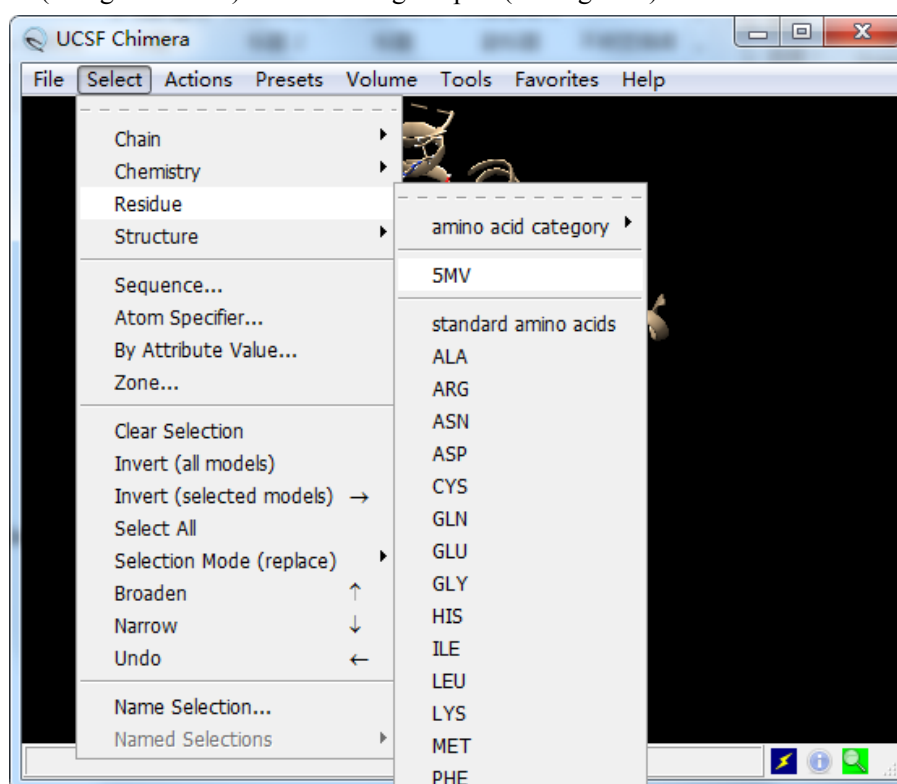


Figure 6. Selecting ligand 5MV.

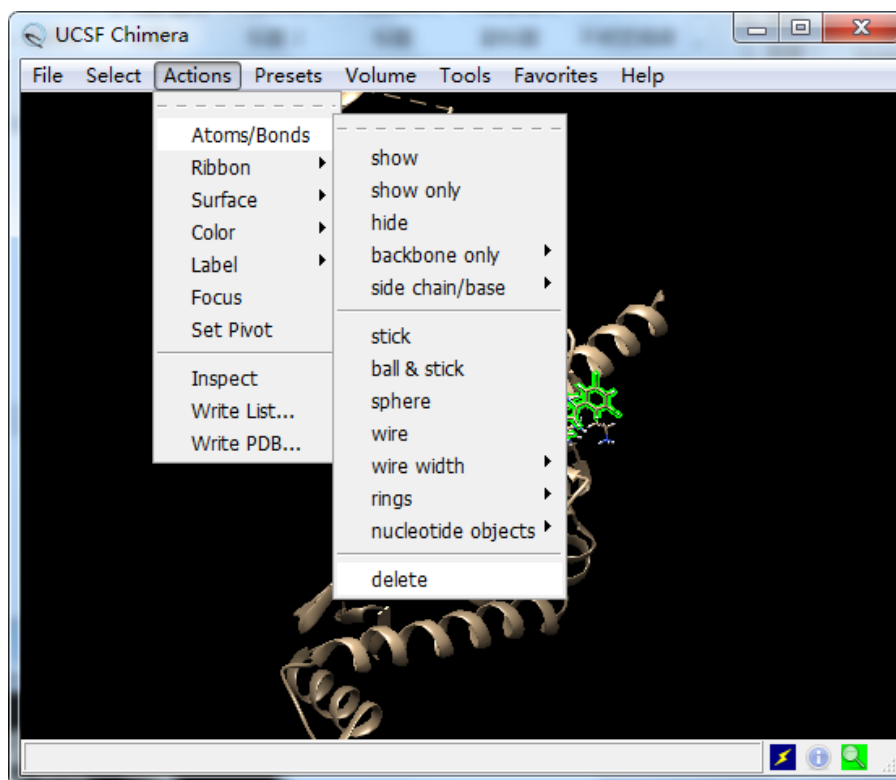


Figure 7. Deleting ligand 5MV.

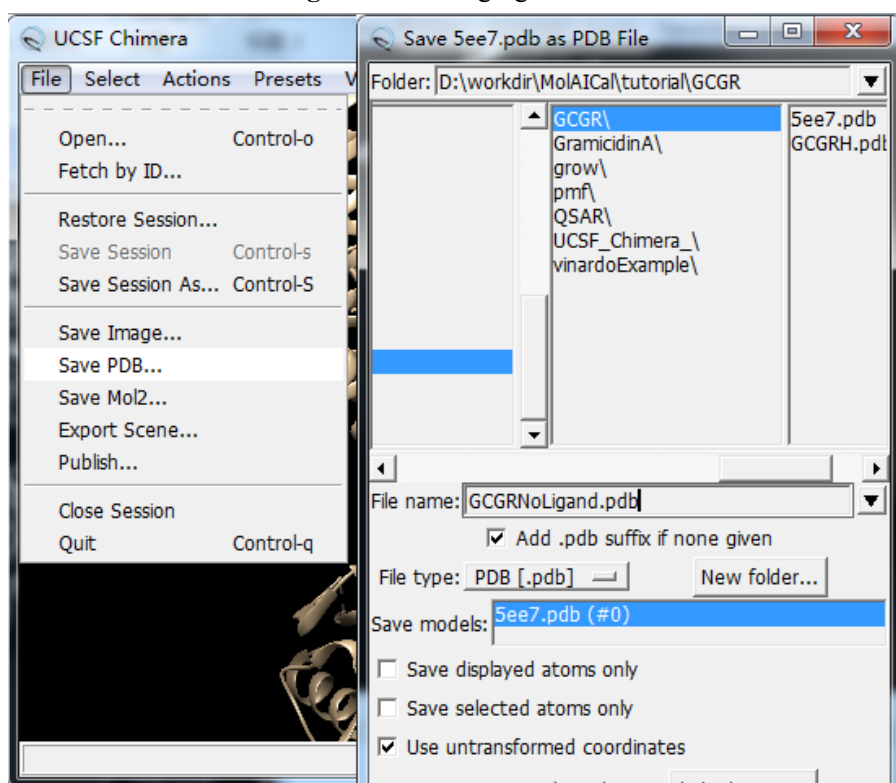


Figure 8. Saving receptor coordinates as PDB file.

3.2. Dealing with ligand

1. Close session (File→Close Session) and reload the GCGRH.pdb, then select with ligand SUV (see Figure 9 and 10):

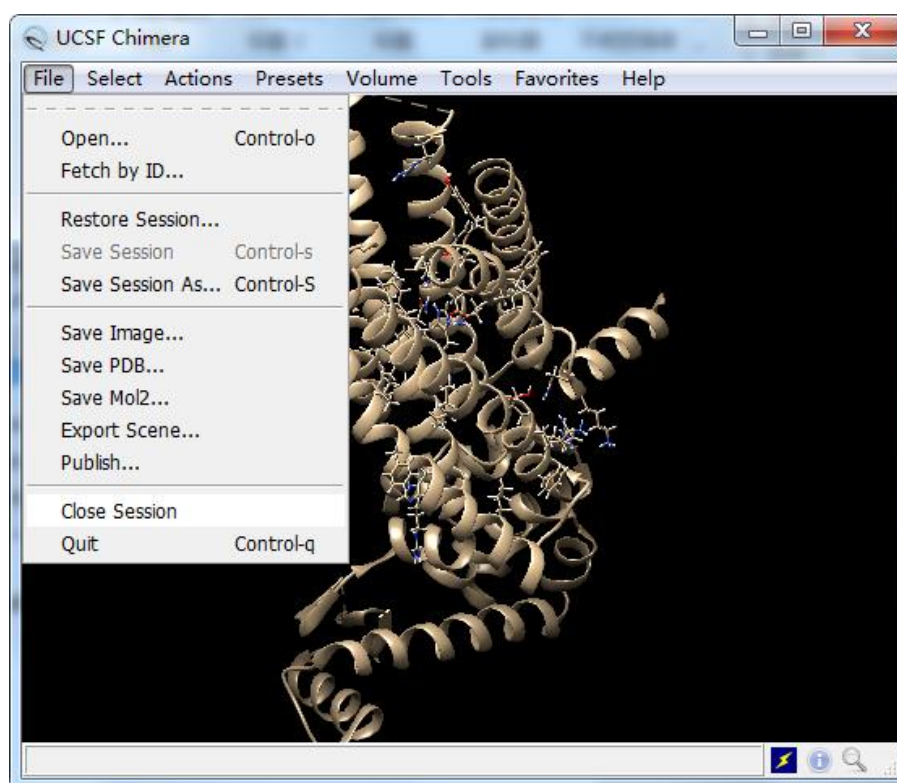


Figure 9. Closing session.

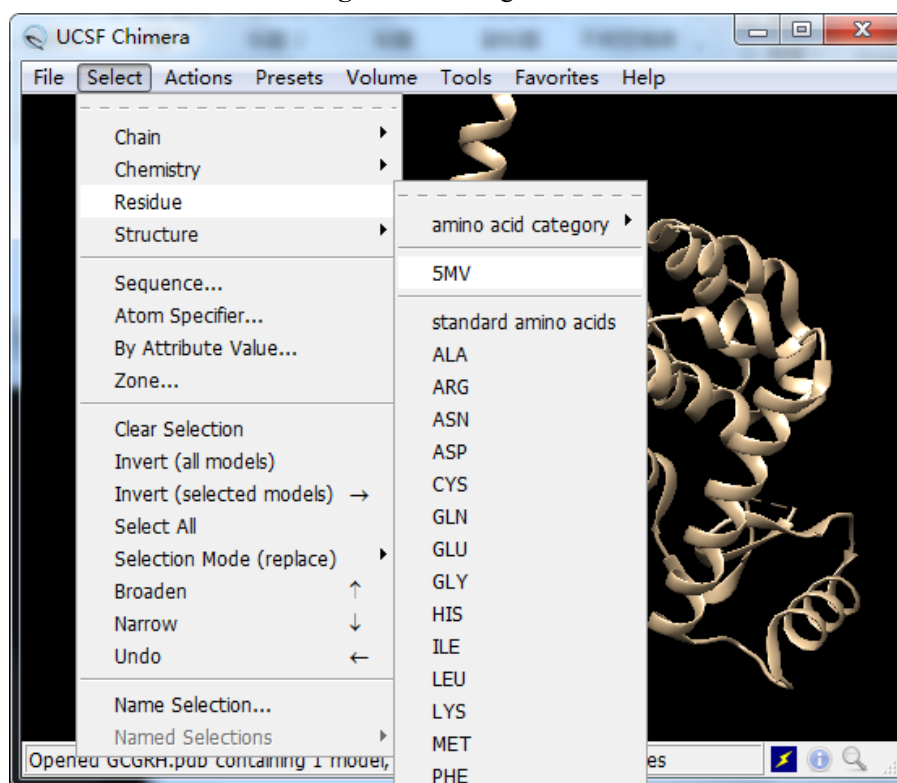


Figure 10. Selecting ligand 5MV.

2. Invert (select models): invert select ligand 5MV (see Figure 11):

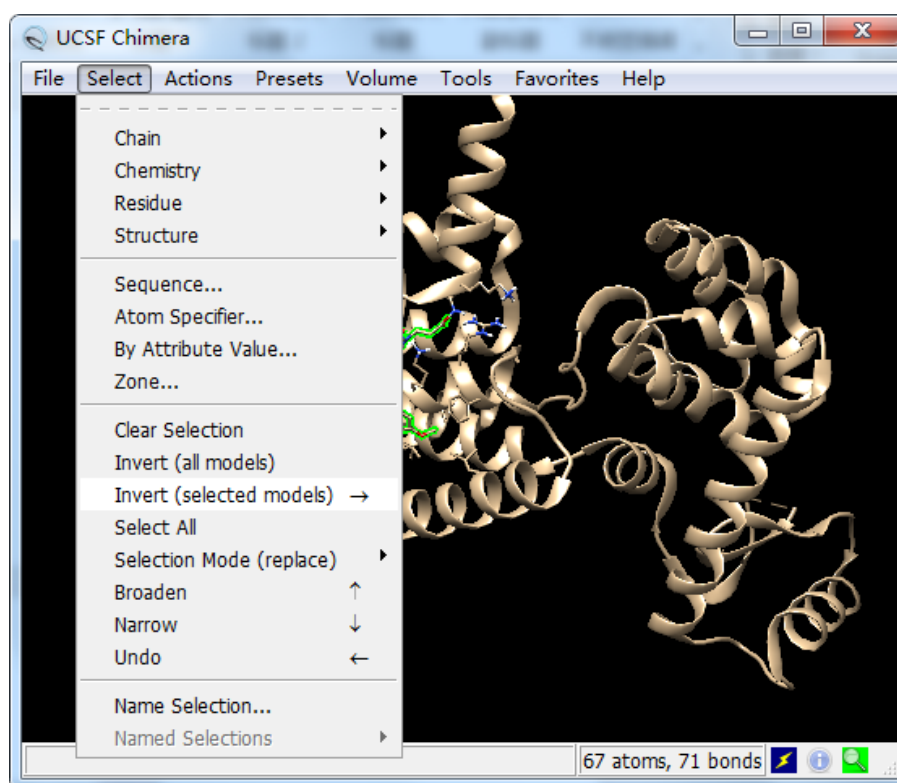


Figure 11. Inverting selected ligand 5MV.

3. Then delete invert selection (see Figure 12):

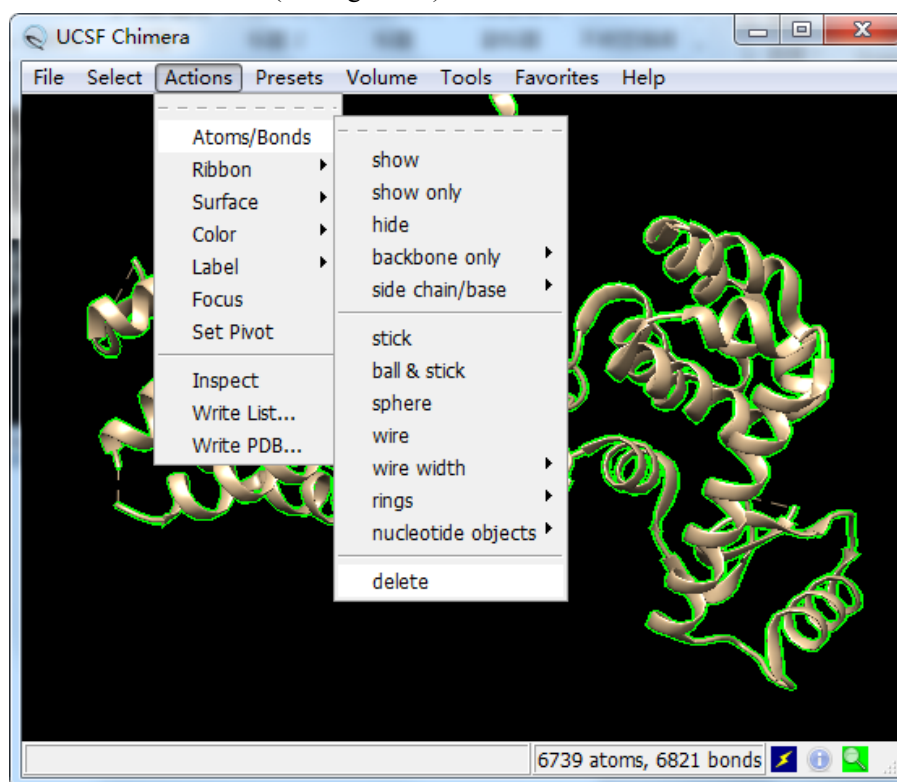


Figure 12. Deleting receptor.

4. Then save ligand to ligand.pdb. It can be split into different fragments which can be chosen as the start growth fragment in the pocket of GCGR (see Figure 13).

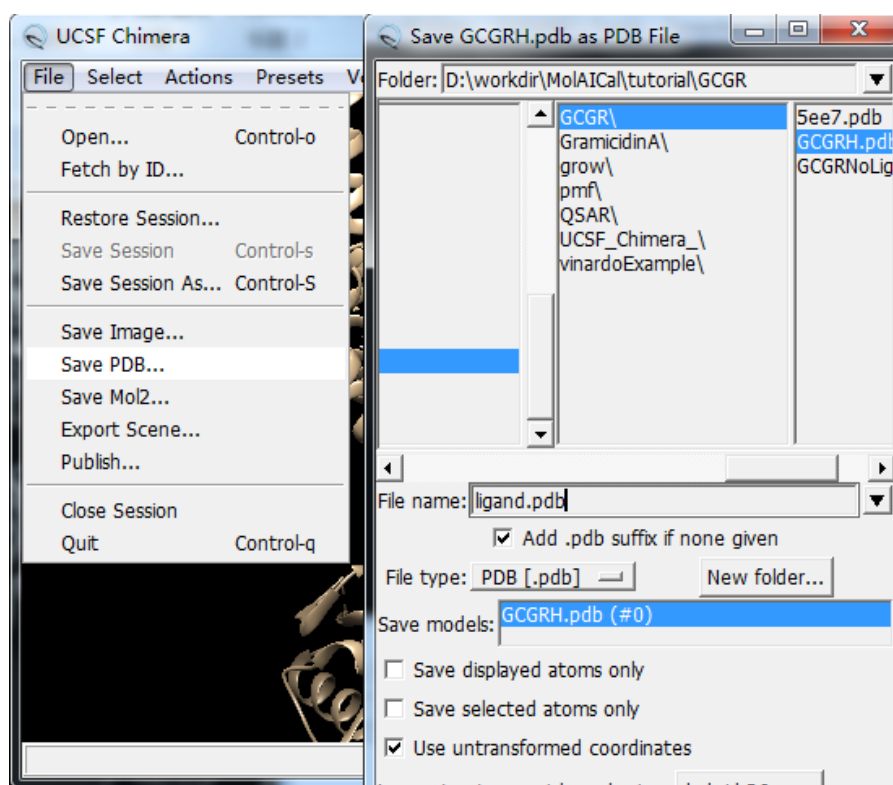


Figure 13. Saving ligand coordinate.

3.3. Calculating center of growth box

Calculating the geometry center of growth box. (Tip: If no ligand, you can choose the key residues as experiment report.)

1. Close session (File→Close Session) and reload the GCGRH.pdb, then select the ligand (see Figure 14):

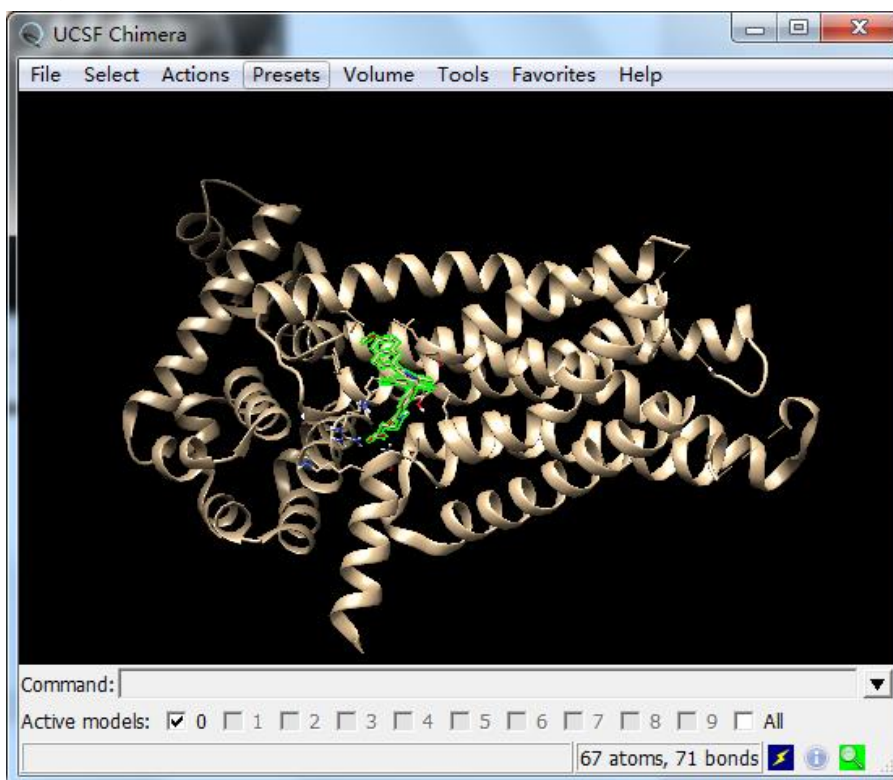


Figure 14. Saving ligand coordinates.

2. Tools→Structure Analysis→Distance (see Figure 15):

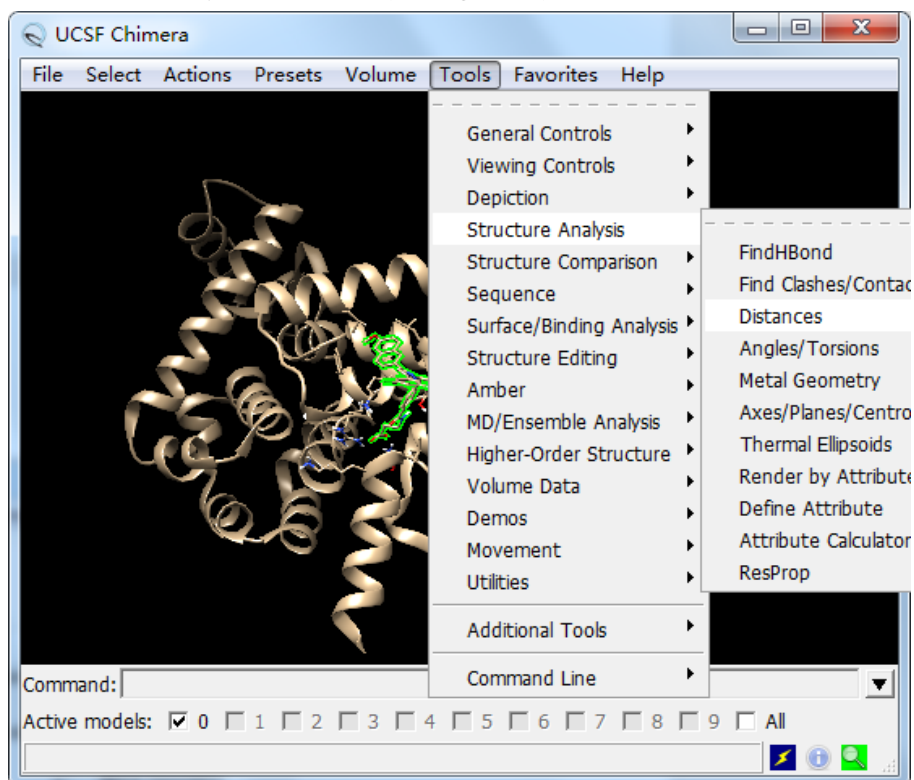


Figure 15. Selecting distance tool.

3. Click Axes/Planes/Centroids, then click “Define centroid” (see Figure 16):

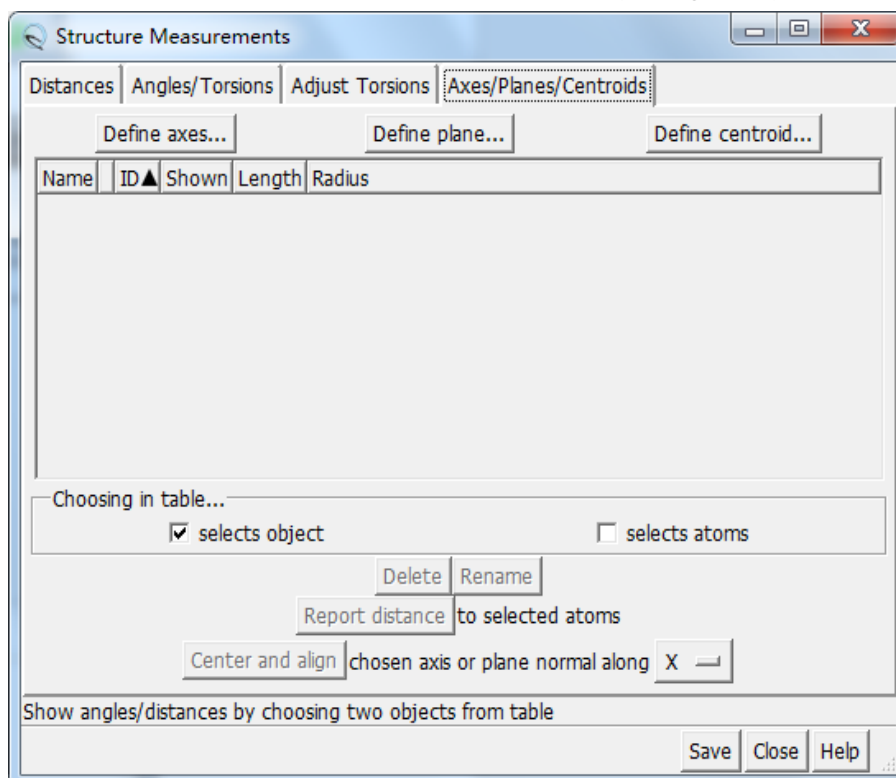


Figure 16. Define centroid.

4. Click “OK” in the pop-up (see Figure 17).

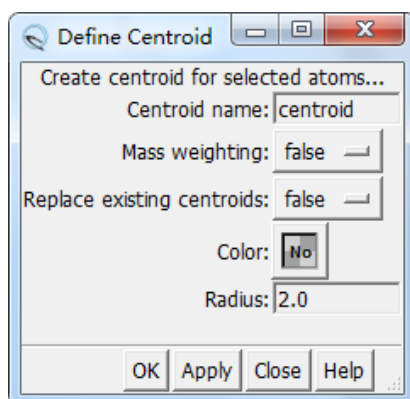


Figure 17. Define the centroid box.

5. Select the defined centroid (blue part) and click “Report distance” (see Figure 18)

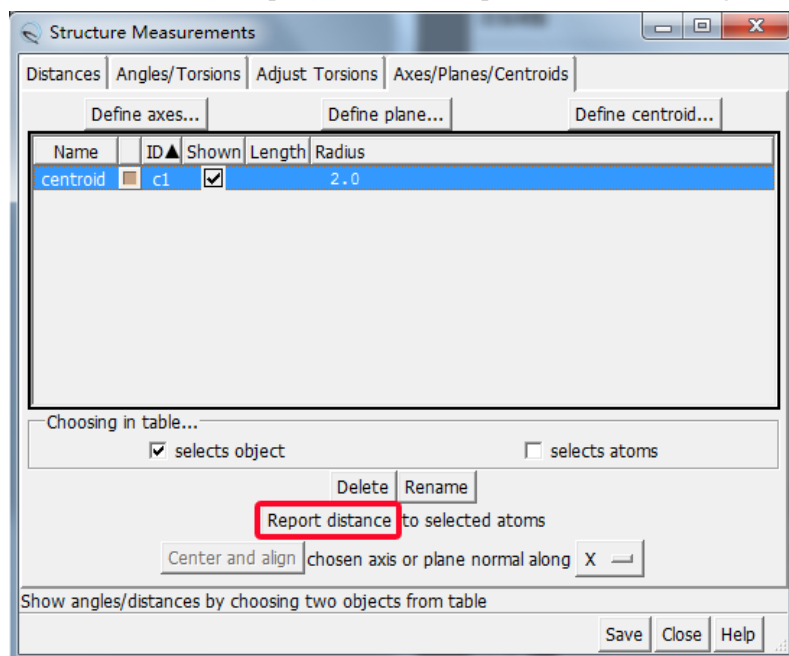


Figure 18. Report distance.

6. Then you can see the center centroid (x, y, z): -30.011, 1.665, -36.581 (see Figure 19):

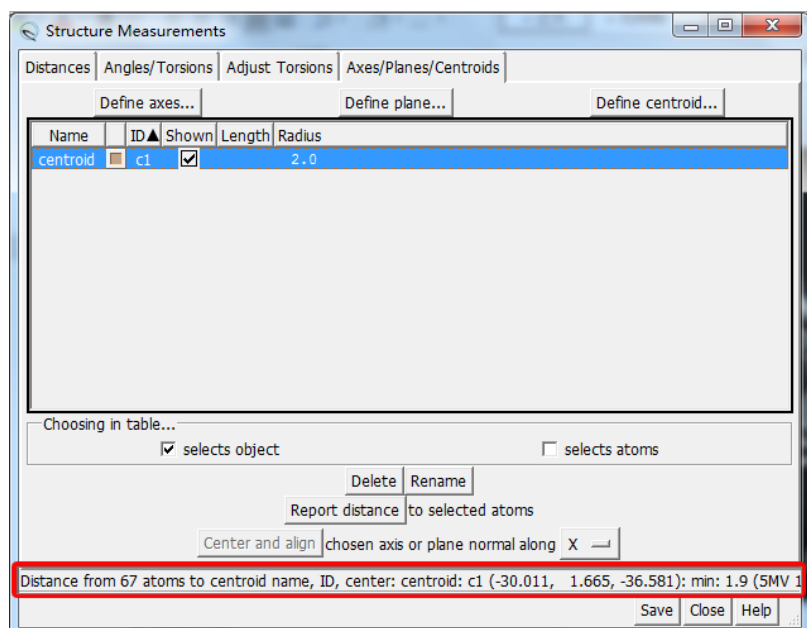


Figure 19. Showing coordinate values of centroid box.

7. Calculating the final box size. You can try X, Y, Z, lengths of 30, 30, 30. Generate the “**box.bild**” by using command of MolAICal as below (note: **the double quotes are necessary for X, Y, Z coordinates. The interval distance between X, Y, Z coordinates should be one space.**):
 - 1) MolAICal.exe -tool box -i "-30.011 1.665 -36.581" -l "30.0 30.0 30.0" -o "D:\workdir\MolAICal\tutorial\GCGR\box.bild"
 - 2) File→open, then open “**box.bild**” (see Figure 20), and check whether the generated box is suitable (see Figure 21).

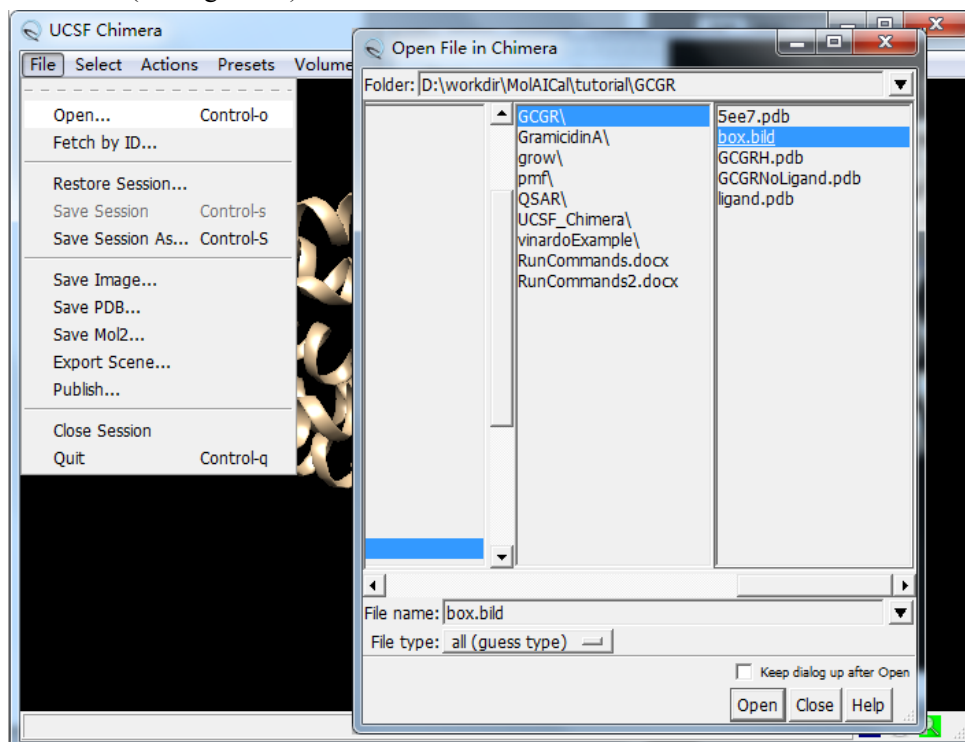


Figure 20. Opening box.bild.

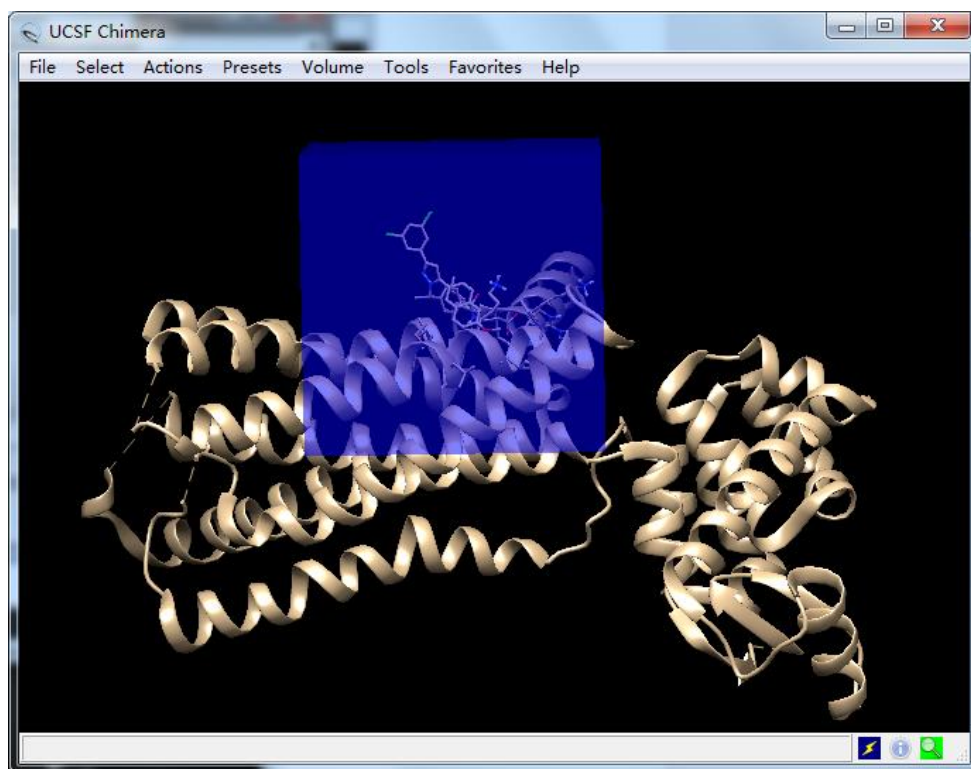


Figure 21. Showing the box in the pocket of GCGR.

The box size of 30, 30, 30 is suitable, so the final center parameter is -30.011, 1.665, -36.581 and the final box lengths of X, Y, Z are 30.0, 30.0, 30.0.

3.4. Making the initial growth fragment

Here, two options are introduced. You can choose anyone of options for de novo drug design. We recommend option 1 if the crystal ligand is determined in the pocket of the receptor.

Option 1. Fragment extracted from crystal ligand

In this option, the initial fragment is extracted from the crystal ligand in the pocket of GCGR.

1. Make the initial growth fragment by UCSF Chimera. Open ligand.pdb saved above. You chose atom via mouse and keyboard:
 - 1) Ctrl + left key of mouse: it can select one atom at a time.
 - 2) Ctrl + Shift + left key of mouse: it can select many atoms.
 In this tutorial, we selected intimal atoms as the initial growth fragment by dragging the Ctrl + left key of mouse (see Figure 22).

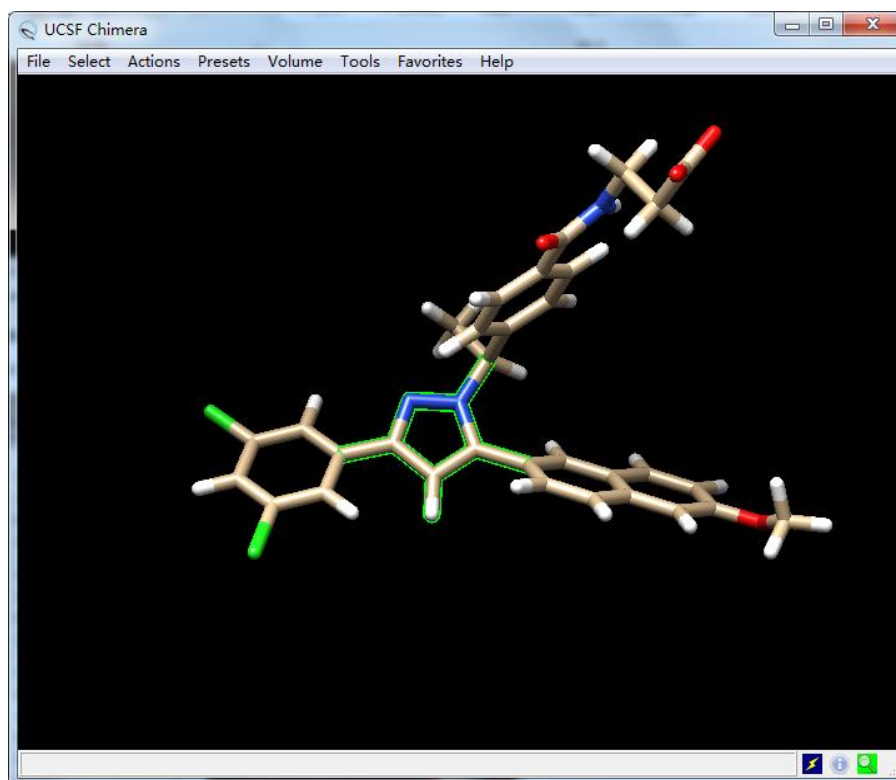


Figure 22. Selecting initial growth part.

2. Invert (selected models) as shown in Figure 23:

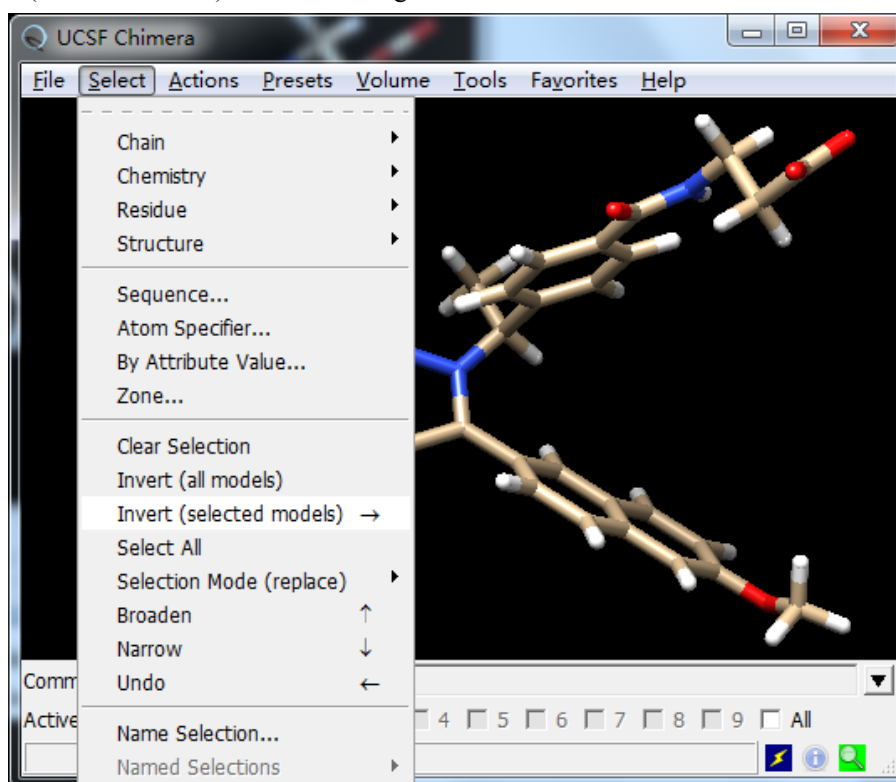


Figure 23. Invert selecting the initial growth part.

3. Then delete the invert selected part: Actions→Atoms/Bonds→delete.
Finally, we choose the below molecule as the starting fragment (see Figure 24):

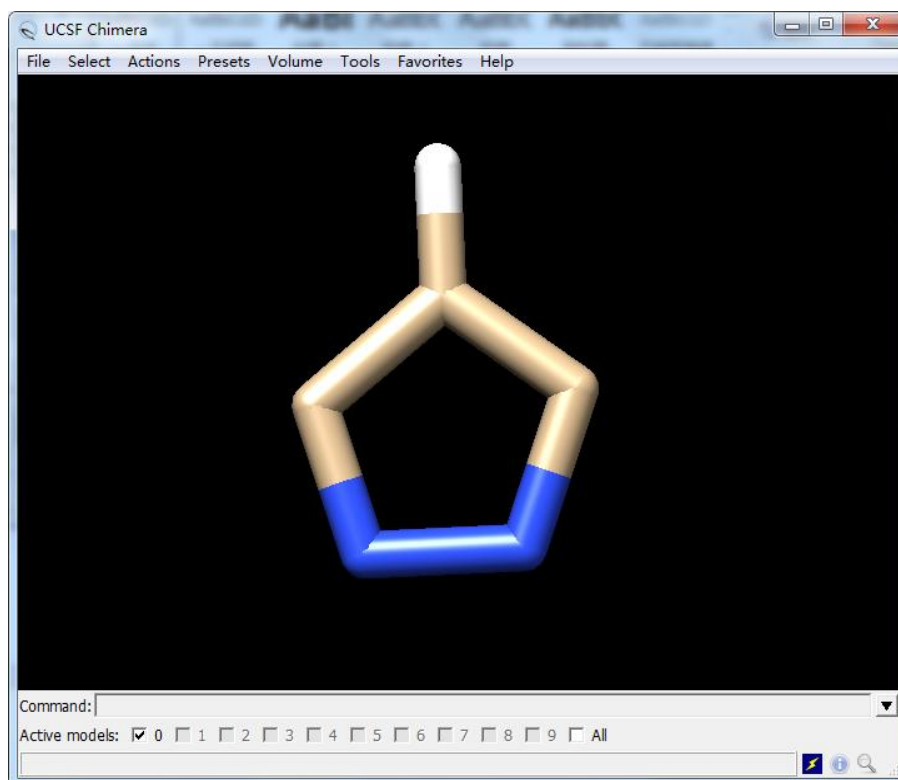


Figure 24. Initial growth fragment.

4. To let the molecule grow in the right way, the initial fragment part should add hydrogen, so “Tools→Structure Editing→AddH” is performed (see Figure 25).

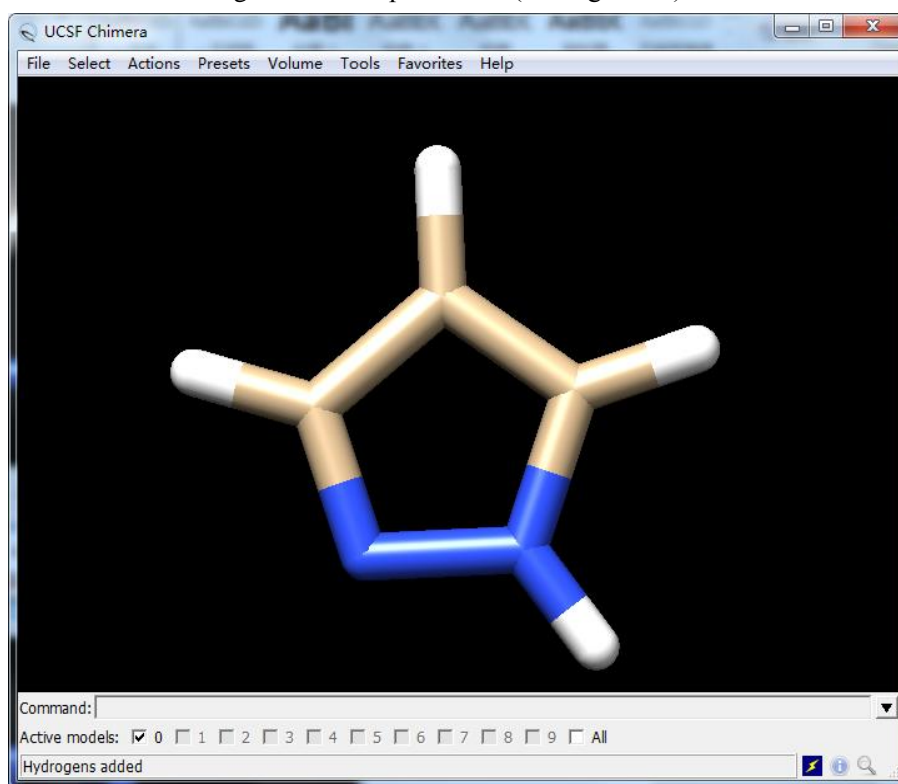


Figure 25. Adding hydrogen in the initial growth fragment.

5. Then, save this fragment as sybyl Mol2 format named “startFrag.mol2” (see Figure 26).

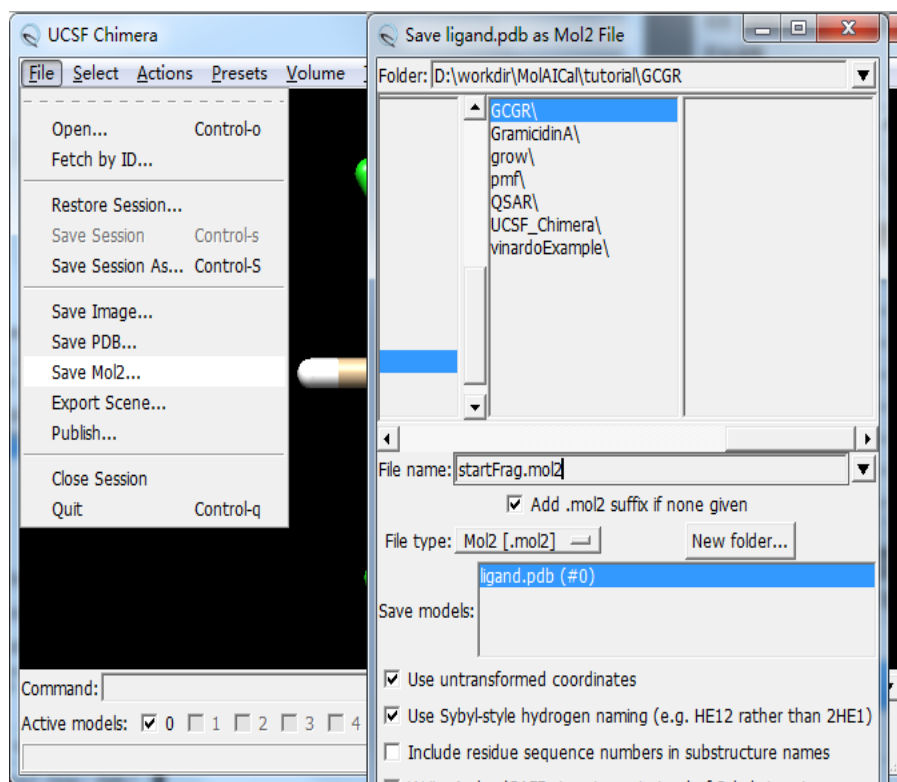


Figure 26. Saving coordinate of the initial growth fragment.

Option 2. No crystal ligand in the pocket.

If there is no crystal ligand in the pocket of receptor, you can choose an atom of the key residues according to the literature reported or your experience by same procedure of option 1. And make this atom as the independent file and give an initial fragment with SMILES format. MolAICal will search the best position of SMILES format fragment in the pocket of GCGR automatically.

Set “growMethod” to randomFrag. Then add parameters “startAtomPosition” and “startSmiFrag”. The “startAtomPosition” contains one atom that is from your selected residues. The “startSmiFrag” is your appointed initial fragment with SMILES format. The parameter instance is as below:

growMethod	randomFrag
startFragFile	D:/GCGR/genstartFrag.mol2
startAtomPosition	D:/GCGR/resPosition.pdb
startSmiFrag	C

Here, the **option 1** is chosen for drug design.

3.5. Running the de novo drug design

There are two options: Option 1 is running de novo drug design by AI model and classical programming. Option 2 is running de novo drug design by pure classical programming. To save the beginner’s time, you can try any below option for your learning.

Option 1. De novo drug design by AI model and classical programming.

Set “libStyle” to AIFrag. The instance is as below:

```
-----  
# define read library way: mol2, SMILES, AIFrag  
libStyle                      AIFrag  
-----
```

Note: if it show the err information: “*Warning: Atom 9 C.3 overlaps with protein!*” and “*Warning: some atoms overlap with protein.*” You can minimize the receptor-ligand complex by MD simulations tools or UCSF Chimera, etc. Only the win64 or linux64 version of MolAICal can perform in this option.

Change console to the directory “001-AIGrow” according to the specific position of tutorial material.

```
#> cd 001-AIGrow
```

Finally run the following command in the background:

For linux:

```
#> molaical.exe -denovo grow -i InputParFileAI.dat >& denovo.log &
```

For windows (using PowerShell):

```
#> molaical.exe -denovo grow -i InputParFileAI.dat
```

If you want to run it background, you can run below command:

```
#> powershell -windowstyle hidden -command “molaical.exe -denovo grow -i InputParFileAI.dat”
```

Option 2. De novo drug design by classical programming

Set “libStyle” to mol2. The MolAICal will use the user defined libraries. The instance is as below:

```
-----  
# define read library way: mol2, SMILES, AIFrag  
libStyle                      mol2  
-----
```

Note: if it show the err information: “*Warning: Atom 9 C.3 overlaps with protein!*” and “*Warning: some atoms overlap with protein.*” You can minimize the receptor-ligand complex by MD simulations tools or UCSF Chimera, etc. Any version of MolAICal can perform in this option.

MolAICal can invoke many CPU cores for de novo drug design by JAVA parallel stream, you should the parameter “coreNum” according to your computer in the input configure file.

Change console to the directory “001-AIGrow” according to the specific position of tutorial material.

```
#> cd 001-AIGrow
```

Finally run the following command in the background:

For linux:

```
#> molaical.exe -denovo grow -i InputParFileCP.dat >& denovo.log &
```

For windows (using PowerShell):

```
#> molaical.exe -denovo grow -i InputParFileCP.dat
```

If you want to run it background, you can run below command:

```
#> powershell -windowstyle hidden -command "molaical.exe -denovo grow -i InputParFileCP.dat"
```

4. Results

You can find the directory “results” when the program finishes. In this tutorial, the generated cycle is set to 30. It needs about 1~2 day by using 30 CPU cores. If you try to see the results, please open the “results” directory of 001-AIGrow. Then you can find a file named “AstatisticsFile.dat” which contains the drug design information. It is just an example that does not contain all results. You can find information in the “AstatisticsFile.dat” as blow:

```
-----  
ID  Name  Cluster  Affinity(kcal/mol) Formula  InChIKey  Synthetic_Accessibility  
1   lig_1.mol2 [1]  -3.27  C9H13N6O14S2  BEQXRJFDXZCPLY-GRQBKTHUSA-O  77.41  
2   lig_2.mol2 [1]  -7.67  C13H13N11O9S  IQOZHKOALGXOOC-WVXRZKCLSA-O  75.31  
.....  
-----
```

The “Affinity” is the binding score. The “Cluster” represents the k-means cluster results. You can pick up the representative ligands for your research. Synthetic_Accessibility represents a score that ranges from 0 to 100. The value 100 is maximal synthetic accessibility which means this compound is most easily synthesizable. For example, load “GCGRNoLigand.pdb”, “ligand.mol2” and “lig_2.mol2” by UCSF Chimera (see Figure 27). The ligand colored with red is “lig_2.mol2”. It has grown the analog of GCGR crystal ligand named “ligand.mol2”.

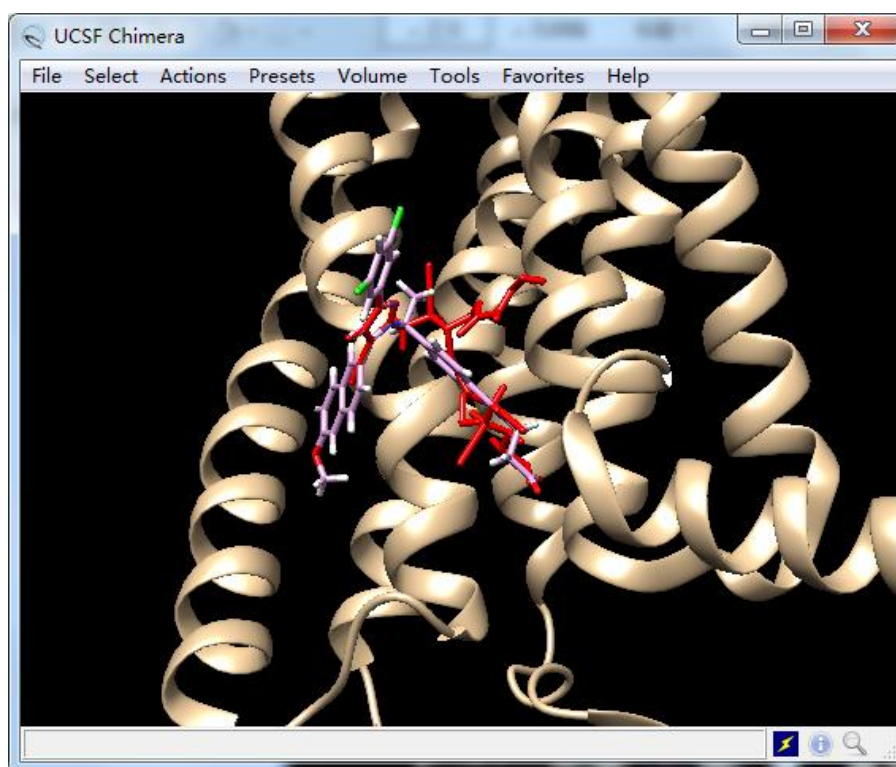


Figure 27. Results of drug design.

Notice: You may find weird hydrogen on hydroxyl (OH group). It may be due to the weird fragments generated by the deep learning model. **This hydrogen is not considered into the binding score. You can delete the hydrogen by UCSF Chimera, and then add hydrogen again.** To evaluate the binding ability of this ligand precisely, the molecular dynamics simulations are recommended.