使用 MolAICal 进行分子能量最小化的教程

Qifeng Bai

邮箱：molaical@yeah.net

个人主页：https://molaical.github.io/baiqf.html

1. 引言

有时，受体口袋中对接的配体会出现轻微的构象扭曲，以及一些研究者出于特定目的需要优化配体结构。分子能量最小化能提供一种有效的方法，可以用于优化分子。MolAICal 提供了两种分子能量最小化方法：一种是 仅使用 MolAICal 实现分子能量最小化，另一种是基于 MolAICal 的接口与 UCSF Chimera 实现的。有关分子能量最小化的更多介绍，请参阅 MolAICal 手册。

2. 材料

2.1 软件要求

1. MolAICal：https://molaical.github.io 或 https://molaical.gitlab.io
2. UCSF Chimera：https://www.cgl.ucsf.edu/chimera

注意：确保 MolAICal 安装正确！

2.2 示例文件

1. 所有必要的教程文件可从以下地址下载：

https://gitee.com/molaical/tutorials/tree/master/025-minimization

本教程介绍两种不同的能量最小化方法，分别在第一部分和第二部分详细阐述。读者可根据自身需求选择阅读其中一部分或按顺序阅读两部分。

第一部分：通过 MolAICal 进行能量最小化

进入 “025-minimization” 文件夹，该文件夹包含一些蛋白质文件（GCGRH.mol2 和 GCGRH.pdb）、配体文件（lig_10.mol2 和 lig_10.pdb）以及名称含 “_fix” 的文件（用于指定能量最小化的固定原子）。

1. 检查 1-2 个分子信息，以确认输入分子的原子序号：

```
#> molaical.exe -min1 -i lig_10.mol2
#> molaical.exe -min2 -m info -i1 GCGRH.pdb -i2 lig_10.pdb
```

2. 单分子最小化：

✧ 无固定原子的单分子最小化

```
#> molaical.exe -min1 -i lig_10.mol2 -o min_lig.mol2
```

✧ 带固定原子序号的单分子最小化

```
#> molaical.exe -min1 -i lig_10.mol2 -o min_lig.mol2 -f 1-2
```

✧ 带参考固定分子文件、快速优化迭代次数和精修迭代次数的单分子最小化

```
#> molaical.exe -min1 -i lig_10.mol2 -o min_lig.mol2 -f lig_fix.mol2 -q 10 -r 6
```

3. 双分子最小化，仅输出第二个输入分子的结果：

✧ 双输入分子最小化

```
#> molaical.exe -min2 -m 1 -i1 GCGRH.pdb -i2 lig_10.pdb -f1 GCGRH_fix.pdb -f2 0 -o2
min_lig_1.pdb
```

✧ 将第二个输入 PDB 分子转换为 Mol2 格式并进行双分子最小化

```
#> molaical.exe -min2 -m 1 -i1 GCGRH.pdb -i2 lig_10.pdb -f1 GCGRH_fix.pdb -f2 0 -o2
min_lig_2.pdb -c2
```

分析：若 “-f2” 为 0，表示第二个分子无固定原子。打开配体文件：lig_10.pdb、min_lig_1.pdb 和 min_lig_2.pdb。**由于 PDB 格式的分子没有键连接信息，进行能量最小化时部分键会发生变化（见图 1）**。而 mol2 格式的分子包含键连接信息，因此不存在键缺失问题。对于固定原子，键连接性保持不变，因此，固定的 PDB 格式分子可直接用于能量最小化，无需考虑键参数。“-c2” 选项会将第二个分子（PDB 格式）转换为 mol2 格式，然后对转换后的结构进行能量最小化。

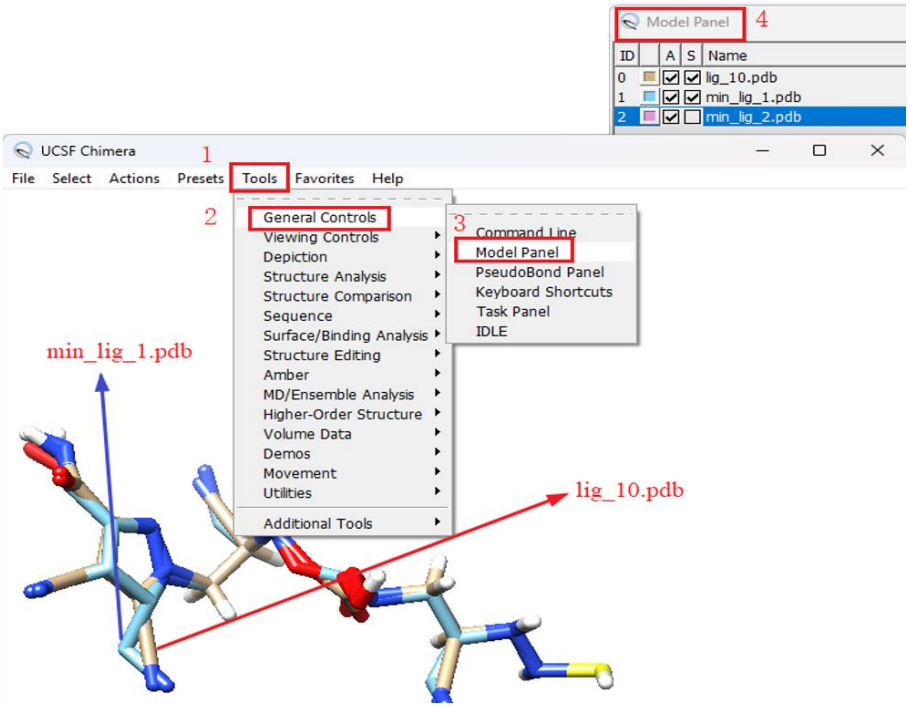

图 1

4. 双分子最小化，输出两个输入分子的结果：

✧ 双输入分子最小化

```
#> molaical.exe -min2 -m 2 -i1 GCGRH.pdb -i2 lig_10.pdb -o1 min_gcgr.mol2 -o2 min_lig.mol2
```

✧ 将第二个输入 PDB 分子转换为 Mol2 格式并进行双分子最小化

```
#> molaical.exe -min2 -m 2 -i1 GCGRH.pdb -i2 lig_10.pdb -o1 min_gcgr.mol2 -o2 min_lig.mol2 -c2
```

MolAICal 最小化参数说明如下：

-f1： 第一个分子的固定原子（例如，'1-3,5,7-9'）或参考分子文件。

-f2： 第二个分子的固定原子（例如，'1-3,5,7-9'）或参考分子文件。

-h: 打印帮助信息。

-i1： 第一个输入分子文件（.pdb、.mol2、.sdf、.mol 格式）。

-i2： 第二个输入分子文件（.pdb、.mol2、.sdf、.mol 格式）。

-m： 最小化操作模式（默认：null），取值可为 info、1 或 2：“info” 表示仅打印分子信息；“1” 表示仅对一个分子进行最小化；“2” 表示对两个分子进行最小化。

-o1： 第一个分子最小化后的输出文件（.pdb、.mol2、.sdf、.mol 格式）。

-o2： 第二个分子最小化后的输出文件（.pdb、.mol2、.sdf、.mol 格式）。

-q： 快速优化的迭代次数。

-r： 精修的迭代次数。

-c2: 若命令行中包含 “-c”，则将第二个输入 PDB 文件转换为 mol2 格式，该选项为 MolAICal 的分子对接功能设计。

第二部分：通过 MolAICal 的接口和 UCSF Chimera 进行能量最小化

若第一部分（Part I） 的能量最小化即可满足研究目标，则可省略本部分内容。若第一部分流程输出的结果有误，第二部分（Part II） 则提供了一套经过验证的备选方案。进入 “025-minimization” 文件夹：

模式 0：可直接运行 UCSF Chimera 的脚本文件

✧ 通过 MolAICal 运行 UCSF Chimera 的命令脚本

```
#> molaical.exe -cmin -c "/home/feng/soft/chimera115/bin/chimera" -m 0 -i1 run.com
```

注意：“-c” 需要指向 “chimera” 程序的路径，“run.com” 是 UCSF Chimera 的命令脚本文件。

模式 1：单分子最小化

✧ 带指定原子序号的单分子最小化

```
#> molaical.exe -cmin -c "/home/feng/soft/chimera115/bin/chimera" -m 1 -i1 lig_10.mol2 -sv "#0:@/serialNumber>=20" -o1 out_1.mol2
```

注意：此处选项 “-i1” 为输入分子文件，与模式 0 不同。在执行在执行能量最小化的过程中，选项 “-sv” 对应的是选择的固定原子，固定原子的选择采用 UCSF Chimera 的方式，详情参见 https://www.rbvi.ucsf.edu/chimera/current/docs/UsersGuide/midas/frameatom_spec.html。**对于模式 1 和 2，**-sv 后面的值可以是：**none**，**selected**，**unselected**，或 atom-spec（UCSF Chimera 中的原子选择方式）。

模式 2：双分子最小化

✧ 带指定原子指定的双分子最小化，并输出两个分子

```
#> molaical.exe -cmin -c "/home/feng/soft/chimera115/bin/chimera" -m 2 -i1 lig_10.mol2 -i2 GCGRH.pdb -sv "#2:<0>" -os "#2:<0>" -o1 out_1.mol2 -o2 out_2.pdb
```

注意：

• 参数 -i1 lig_10.mol2 对应模型标记 #0

• 参数 -i2 GCGRH.pdb 对应模型标记 #1

• 参数 -sv "#2:<0>" 表示选择合并后复合物（#2）中名为"<0>"残基进行固定，再进行能量最小化，此处 #2 即合并 GCGRH.pdb (#1)和 lig_10.mol2 (#0) 的复合物。**对于模式 1 和 2，**-sv 后面的值可以是：**none**，**selected**，**unselected**，或 atom-spec（UCSF Chimera 中的原子选择方式）。

• 参数 -os "#2:<0>" 用于在能量最小化后从复合物（#2）中选择保存其中一个分子（<0>），未被选中的另一个分子将以反向选择的方式保存。**这是因为能量最小化计算是在第一个分子（#0）和第二个分子（#1）合并形成的复合物（#2）上执行的**。

模式 3：含一个固定分子的双分子最小化

✧ 含一个固定分子的双分子最小化，并输出两个分子

```
#> molaical.exe -cmin -c "/home/feng/soft/chimera115/bin/chimera" -m 3 -i1 lig_10.mol2 -i2 GCGRH.pdb -f1 GCGRH_fix.pdb -sv "#0 za<0.01" -os "#3:<0>" -o1 out_1.mol2 -o2 out_2.pdb
```

注意：

- **对于模式 3 和 4，**-sv 后面的值必须是 atom-spec（UCSF Chimera 中的原子选择方式）和区域选择器（Zone specifiers）。
- **在模式 3 中，**参数 -sv 的默认值为 "#0 za<0.01"。
- **区域选择器**（Zone specifiers）用于定义与参考原子相距特定范围内的原子和残基。操作符 z< 和 zr< 会选择所有包含至少一个原子位于给定距离内的残基，而 za< 则直接选择该距离内的所有独立原子。互补操作符 z>、zr> 和 za> 则返回与对应 < 操作符相反的结果集。例如：

  #1:UTK za<12.5

  表示选择模型 #1 中 UTK 残基任意原子 12.5 Å 范围内的所有原子。

模式 4：含两个固定分子的双分子最小化

✧ 含两个固定分子的双分子最小化，基于 MolAICal 的完整选项输出一个分子和日志文件

```
#> molaical.exe -cmin -c "/home/feng/soft/chimera115/bin/chimera" -m 4 -i1 lig_10.mol2 -i2 GCGRH.pdb -f1 GCGRH_fix.pdb -f2 lig_fix.mol2 -os "#4:<0>" -cm gas -fs freeze -sv "#0,1 za<0.01" -fv false -pv false -ns 10 -ss 0.02 -cs 2 -css 0.02 -iv 1 -ng true -mf1 mol2 -mf2 pdb -o1 out_1.mol2 -o2 out_2.pdb -w false -p false
```

注意：

- **对于模式 3 和 4，**-sv 后面的值必须是 atom-spec（UCSF Chimera 中的原子选择方式）和区域选择器（Zone specifiers）。
- **在模式 4 中，**参数 -sv 的默认值为 "#0,1 za<0.01"。
- **区域选择器**（Zone specifiers）用于定义与参考原子相距特定范围内的原子和残基。操作符 z< 和 zr< 会选择所有包含至少一个原子位于给定距离内的残基，而 za< 则直接选择该距离内的所有独立原子。互补操作符 z>、zr> 和 za> 则返回与对应 < 操作符相反的结果集。例如：

  #1:UTK za<12.5

表示选择模型 #1 中 UTK 残基任意原子 12.5 Å 范围内的所有原子。

UCSF Chimera 与 MolAICal 的最小化参数说明如下：

-c,--chimera_path：　　Chimera 可执行文件的路径

注意：MolAICal 还提供了一种设置 Chimera 路径的方法，请注意 MolAICal 在 Windows 和 Linux 版本中的配置有所不同：**Linux 版本采用 udocker 容器方法，需要在容器内进行设置，**而 Windows 版本则无需如此。对于 Linux 版本 MolAICal 中 UCSF Chimera 的配置或安装，用户可以参考**附录 1** 中的 NAMD 安装步骤。MolAICal 的 Linux 和 Windows 版本在配置 UCSF Chimera 的最后一步非常相似，如下：

```
#> molaical.exe -call set -n chimera -p "root/soft/Chimera119/bin/chimera"
```

参数 -n 的值必须为 chimera（不区分大小写）。目前，VMD、NAMD 和 chimera 是 MolAICal 内部路径指定的固定名称，因此必须严格按此书写，但字母大小写不影响。-p 需指定 UCSF Chimera 可执行程序的绝对路径。

设置成功后，该路径将被自动保存并调用，**无需每次重复输入 chimera_path，从而减少冗余操作**。因此，在执行能量最小化时，无需再输入 -c 或 --chimera_path 参数。例如：针对前文提到的模式 3（Mode 3），命令可简化为如下格式：

```
#> molaical.exe -cmin -m 3 -i1 lig_10.mol2 -i2 GCGRH.pdb -f1 GCGRH_fix.pdb -sv "#0 za<0.01" -os "#3:<0>" -o1 out_1.mol2 -o2 out_2.pdb
```

-cm,--charge_method：电荷计算方法（am1 或 gas），默认：gas

-cmin：来自 UCSF Chimera 接口的最小化计算

-cs,--cgsteps_value：共轭梯度（CG）步数，默认：2

-css,--cgstepsize_value：共轭梯度（CG）步长，默认：0.02

-f1,--fixed_f1：第一个固定分子文件的路径（用于模式 3 和模式 4）

-f2,--fixed_f2：第二个固定分子文件的路径（用于模式 4）

-fs,--freeze_spec：冻结或指定方法，默认：freeze

-fv,--fragment_value：是否使用片段（true/false），默认：false

-h,--help：显示帮助信息

-i1,--input_f1：第一个输入分子文件或命令文件的路径（用于模式 0）

-i2,--input_f2：第二个输入分子文件或输出日志的路径（用于模式 0）

-iv,--interval_value：间隔，默认：1

-m,--mode：操作模式（0、1、2、3 或 4）：

模式 0：直接调用 chimera_command（--input_f1 作为命令文件，--input_f2 作为输出日志）；

模式 1：单分子最小化；

模式 2：双分子最小化；

模式 3：含一个固定分子的三分子最小化；

模式 4：含两个固定分子的四分子最小化。

-mf1,--mol_format_1：第一个输出文件格式，默认：mol2

-mf2,--mol_format_2：第二个输出文件格式，默认：pdb

-ng,--nogui_value：是否使用 GUI（true/false），默认：true

-ns,--nsteps_value：最速下降步数，默认：10

-o1,--output_f1：第一个输出文件

-o2,--output_f2：第二个输出文件

-os,--output_sel：输出选择

-p,--print_output：是否打印输出（true/false），false 表示输出日志文件，反之亦然，默认：true

-pv,--prep_value：是否使用 prep（true/false），默认：true

-ss,--stepsize_value：最速下降步长，默认：0.02

-sv,--sel_value：原子的选择值

-w,--whether_output：是否输出第二个分子 (true/false)，false 表示不输出第二个分子，反之亦然，默认：true

附录 1

在 MolAICal 容器内安装外部程序（推荐，适用于 Linux 版本的 MolAICal）

请注意，Windows 版与 Linux 版的 MolAICal 配置存在差异：Linux 版本采用基于 udocker 容器的技术方案，需在容器内部完成设置，而 Windows 版本则无需此步骤。

1. 首先，将文件复制到 MolAICal 容器中

```
# 进入容器文件系统（将进入 "/root" 目录）
#> molaical.exe -eset shell in

# 将 VMD 和 NAMD 安装包从本地机器复制到容器中，'cp' 命令的第一部分（源路径）
# 位于本地主机，第二部分（目标路径）在容器内；VMD 和 NAMD 软件包可通过以下命
# 令移入容器。
#> cp /home/user/<本地文件> /root/soft

# 退出容器文件系统
#> exit
```

2. 其次，进入 MolAICal 容器的虚拟环境

```
#> molaical.exe -eset sys run molaical
```

注：molaical 是容器名称。

3. 在 MolAICal 容器虚拟环境中安装软件的方式与在本地主机上安装相同。以下以安装 VMD 和 NAMD 为例：

1) 安装 NAMD：

解压 NAMD 文件（假设解压后的文件夹名为 namdcpu），然后使用以下命令将其路径告知 MolAICal：

```
#> molaical.exe -call set -n NAMD -p "/root/soft/namdcpu/namd3"
```

注：请将上述 VMD 和 NAMD 的路径替换为您系统中的实际路径。-n 后面的 "VMD" 和 "NAMD"（大小写不敏感）是固定的标识符。为确保 MM/GBSA 结果的可重复性，建议使用 NAMD 的 CPU 版本，因为 CUDA 版本中的 seed 参数似乎对结果可重复性无效。

2) 安装 VMD：

按以下步骤操作：

◆ **解压 VMD 文件：**

```
#> tar -xzvf vmd-xxx.tar.gz
```

注：请将上述路径替换为您系统中的实际路径。

◆ **修改 VMD 解压目录中名为 configure 的文件中的安装路径：**

```
# 默认值：
$install_bin_dir="/usr/local/bin";
$install_library_dir="/usr/local/lib/$install_name";

# 修改为：
$install_bin_dir="/root/soft/vmd193/bin";
$install_library_dir="/root/soft/vmd193/lib/$install_name";
```

◆ **安装 VMD：**

```
#> cd vmd-xxx
#> ./configure LINUXAMD64
#> cd src
#> make install
```

注：请运行 `./configure` 并根据所用计算机选择正确的类型，此处为 `"LINUXAMD64"`。

◆ **然后使用以下命令将 VMD 路径告知 MolAICal：**

```
#> molaical.exe -call set -n VMD -p "/root/soft/vmd193/bin/vmd"
```

至此，NAMD 和 VMD 在 MolAICal 容器内的安装与配置已完成。

✧ **记得使用 exit 命令退出 MolAICal 虚拟环境，返回本地计算机进行计算（主要是为了省去文件拷贝步骤；在容器内运行也可行，但需手动将数据从本地计算机复制到 MolAICal 容器中）。**

```
#> exit
```

3) 保存并加载已修改的容器（可选）

为保留容器内所做的更改，并避免日后重新安装软件（因为一旦容器被删除，所有数据都会丢失），可执行以下操作：

✧ **获取容器 ID 和名称：**

```
#> molaical.exe -eset sys ps
```

✧ **克隆该容器：**

```
#> molaical.exe -eset sys export --clone -o molaicalv2.tar molaical
```

注：`molaical` 是容器名称，也可使用容器 ID。如果报错，那可能是因为挂载的文件系统的权限问题，可以忽略。

✧ **假如需要，导入克隆的容器：**

```
#> molaical.exe -eset sys import --clone --name molaicalv2 molaicalv2.tar
```

✧ **将新导入的容器名称更改为默认容器名 `"molaical"`，原容器重命名为 `"bakmolaical"`：**

```
#> molaical.exe -eset sys rename molaical bakmolaical
#> molaical.exe -eset sys rename molaicalv2 molaical
```

注意：容器（动态）是从镜像（静态）生成的。软件安装等操作必须在动态容器中进行；如果克隆了该容器，恢复时仍基于原始底层镜像。

更多详情请参阅 MolAICal 用户手册，或运行以下命令获取帮助：

```
#> molaical.exe -eset sys --help
```