

使用 MolAICal 基于 NAMD 模拟结果计算 小分子和蛋白 MM/PBSA 的教程

作者：Qifeng Bai

更多教程（含英文教程）请见如下：

MolAICal 官方主页：<https://molaical.github.io>

MolAICal 官方主页中国镜像：<https://molaical.gitlab.io>

MolAICal 中文博客：<https://molaical.gitlab.io/cntutorial.html>

1. 简介

在本教程中介绍了基于 NAMD 的分子动力学模拟结果，使用 MolAICal 计算小分子和 Mpro 蛋白受体 MM/PBSA 的方法。本教程只是一个简单演示。为了节省运行及存储空间，本教程仅选择了 Mpro 复合物分子动力学模拟的 25 帧用于计算。本教程不仅可以用于计算蛋白质-配体的 MM/PBSA，还可以用于计算基于 MD 模拟的蛋白质-多肽、蛋白质-蛋白质、DNA-配体、DNA-蛋白质、蛋白质-DNA、蛋白质-RNA 和其它任意复合物的 MM/PBSA；只需要用指定的对象替换本教程中的蛋白质和配体即可。例如，基于本教程中相同的运行命令参数，本教程中的蛋白质被替换为 DNA，配体被替换为多肽。

2. 工具

2.1. 所需软件下载地址

1) MolAICal: <https://molaical.github.io> 或 <https://molaical.gitee.io>

2) NAMD: <https://www.ks.uiuc.edu/Research/namd/>

注意：本教程可以使用 **NAMD2.x、3.x 或更高版本**。例如，如果使用 **NAMD 3.x** 版本，则使用命令“**namd3**”替代本教程中的命令“**namd2**”。对于更高版本的 NAMD，用户可以使用与前面示例类似的替换方式。

3) Carma: <https://github.com/glykos/carma> or <https://utopia.duth.gr/~glykos/Carma.html>

4) APBS (<https://apbs.readthedocs.io>) or DelPhi (<http://compbio.clemson.edu/lab/delphisw>)

2.2. 操作示例文件

所有用到的操作教程文件均可在下面的网站下载：

<https://gitee.com/molaical/tutorials/tree/master/020-mmpbsa>

3. 操作流程

第一部分：问题的解决

问题：有些用户使用 MM/PBSA 进行计算得到正值，如下：

https://www.ks.uiuc.edu/Research/namd/mailng_list/namd-l.2020-2021/1295.html

如果用户的研究对象不存在多个分子进入不同镜像的问题，可以跳过这个部分，直接进入 Part II.

可能的解决方案：必须检查一下配体是否一直跟受体在同一个周期性盒子中，如果不是，请使用 VMD PBC 将配体和受体裹进同一个周期性盒子中。可以参考：

https://www.ks.uiuc.edu/Research/vmd/script_library/scripts/pbcwrap/

<https://www.ks.uiuc.edu/Research/vmd/plugins/pbctools/>

这里我提供 2 个命令在 VMD Tk Console 中运行：

```
%> pbc wrap -centersel protein -center com -compound chain -all
```

注：在 VMD 中检查复合物结构，若上述命令能将复合物整体包裹在一起（即：去周期化条件），则可跳过以下命令（pbc unwrap）。

```
%> pbc unwrap -sel "not (water or ions)" -all
```

运行上述命令可以将配体和受体包进同一个周期性盒子。使用下面的命令保存轨迹到 DCD 文件中：

```
%> set all [atomselect top all]  
%> animate write dcd pro.dcd sel $all beg 0 end 24 waitfor all
```

这里所有原子都被选中，用户可以根据需求进行选择。为了简便，本教程只提供 25 帧：从 0 到 24 帧。“pro.dcd”是轨迹名字。解决上述问题之后，可以开始下面的教程。

选项（无图形界面的去周期化条件操作）

若用户已完成去周期化条件操作，或已经在 Linux 版本的 MolAIcal 容器中配置好 VMD 和 NAMD，或无需对 DCD 文件进行去周期化条件操作，可直接跳过此选项步骤。

用户可在 Windows 或 Linux 的图形桌面环境中执行并测试上述命令。但在无图形界面的环境中操作时，需通过命令行输入。为支持外部程序调用，MolAIcal 提供了持久化配置接口。以下命令用于配置 VMD (Visual Molecular Dynamics) 的名称与路径，仅需单次配置即可供后续重复使用，无需重复设置。

为了解决 Linux 中的库依赖问题，Linux 版本的 MolAIcal (Windows 版本的 MolAIcal 没有采用容器)采用了基于容器的方法。如果需要调用外部程序，建议在 MolAIcal 容器内安装这些程序。如果不需要外部程序，可以忽略此步骤。本教程需要外部程序 NAMD 和 VMD，具体步骤如下：

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。

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:

```
#> molaiical.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。为防止 MolAIcal 出现问题时丢失已安装的程序(例如 VMD 和 NAMD), 请参阅附录 1 中的第 3 步。

◇ 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
```

然后, 运行外部程序的参数:

```
#> molaiical.exe -call run -n vmd -i -dispdev text -e pbcwrap.vmd
```

注意:

- 1) 在 MolAIcal 的环境变量配置中, '-n' 后的参数名称(如 '-n VMD')必须与对应的运行时参数(如 '-n vmd')拼写一致, 但大小写可忽略。如上述两个命令中'-n'后的参数名称需保持一致。
- 2) '-i' 后接 VMD 的运行时参数, 且 '-i' 必须置于其他参数之后。此处, 用户只需将 pbcwrap.vmd 替换为自己的脚本文件, 当然也可直接选用本教程提供的脚本 pbcwrap.vmd。下方以 pbcwrap.vmd 为例(用户可根据实际用途或分析结果修改此脚本):

```
package require pbctools
mol new mpro.psf
mol addfile mpro.dcd waitfor all
pbc wrap -centersel protein -center com -compound chain -all
set all [atomselect top all]
animate write dcd mpro.dcd sel $all beg 0 end 24 waitfor all
quit
```

第二部分: 基于 NAMD 轨迹进行 MM/PBSA 的计算

假设用户已在 MolAICal 中配置了 VMD 和 NAMD 路径。上述说明适用于 Linux 版 MolAICal 容器的外部程序安装。对于 Windows 版 MolAICal，用户需将以下命令中的路径替换为实际系统路径后直接运行：

```
#> molaical.exe -call set -n VMD -p "C:\Program Files (x86)\University of Illinois\VMD\vmrd.exe"  
#> molaical.exe -call set -n namd -p "d:\namd2\namd2.exe"  
#> molaical.exe -call set -n delphi -p "E:/delphi/delphicpp_v8.4.3_windows_x64.exe"
```

注意：请将上述 VMD 和 NAMD 的路径替换为用户系统中的实际路径。当运行 MMPB/GB/SA 功能并通过 MolAICal 设置环境变量时（例如：#> molaical.exe -call set -n VMD -p "path"），MMPB/GB/SA 中与"-n"参数对应的变量名是固定的，包括：'vmd'、'namd'和'delphi'（不区分大小写）。

- ◆ 其中 DelPhi 是二进制文件，在 linux 系统上需要赋予可执行权限：`chmod +x delphicpp_v8.4.3_windows_x64.exe`，所以，可以参考[附录 1](#) 中的 NAMD 安装步骤。
- ◆ 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。

3.1 MM/PBSA 的计算

假设已经有跑平衡的轨迹名为"**mpro.dcd**"，同时，在"**mpro.dcd**"轨迹中，保证受体和配体在同一个周期性盒子中。用户可以替换成自己的轨迹。转到以下目录：

```
#> cd 020-mmmpbsa
```

然后运行以下命令，可以快速计算 MM/PBSA：

```
#> molaical.exe -mmmpbsa -f mmmpbsa_apbs.vmd
```

- ◆ 可通过修改文件'mmmpbsa_apbs.vmd'中的参数，或通过输入'-i'参数进行修改。如需更多信息，请查阅 MolAICal 手册。
- ◆ **选项：**此处输入文件'mmmpbsa_apbs.vmd'中的'mpro.dcd'和'mpro.psf'等文件包含蛋白质、配体、离子、水分子等成分，可能会占用大量计算机内存。为节省计算内存，用户可使用以下命令仅提取受体与配体复合物的轨迹：

```
#> molaical.exe -call run -n vmd -c stripDCD -i -dispdev text -psf "mpro.psf" -args protein,or,resname,LIG  
"mpro.dcd" "complex" mpro.psf mpro.pdb
```

注："complex"是输出文件的前缀。系统将参考 mpro.psf 和 mpro.pdb 文件生成以"complex"为前缀的文件，包括：'complex.psf'、'complex.pdb'和'complex.dcd'。这些文件可替代输入文件'mmmpbsa_apbs.vmd'中的'mpro.psf'、'mpro.pdb'和'mpro.dcd'，从而节省加载内存。

运行上述命令后，MM/PBSA 的输出结果和结合自由能 ΔG 如下：

Delta:				
BOND:	25	-0.0000	0.0000	0.0001
ANGLE:	25	-0.0000	0.0000	0.0000
DIHED:	25	-0.0000	0.0000	0.0000
IMPRP:	25	0.0000	0.0000	0.0001
KINETIC:	25	14.0776	0.0000	0.0000
Elec:	25	-30.3701	1.3023	6.5114
Vdw:	25	-52.1546	1.0534	5.2670
PB:	25	23.7231	2.3847	11.9234
SA:	25	-7.6991	0.1004	0.5022
Gas:	25	-82.5247	1.7298	8.6491
Sol:	25	16.0241	2.3622	11.8111

Pol:	25	-6.6470	2.6141	13.0705
Npol:	25	-59.8536	1.1223	5.6116
G binding:	25	-66.5006	+/- 2.7810	13.9052

注: 所有能量值单位均为千卡/摩尔 (kcal/mol)。在不同操作系统或 NAMD 版本中计算结果可能存在差异，但若保留两位小数，结果将保持一致或高度吻合。

输入文件与结果分析

打开 'mmpbsa_apbs.vmd' 文件，请注意这两个参数: "-poisson_boltzmann" 和 "-pb_executable":

- 若用户未设置 "-pb_executable" 参数，MolAICal 将自动调用内置的 APBS 计算泊松-玻尔兹曼 (PB) 能量。
- 若 "-poisson_boltzmann" 设为 2，程序将调用 APBS 计算 PB 能量；设为 1 则调用 DelPhi 计算 PB 能量；设为 0 则执行 MM/GBSA 计算。

若用户将 "-poisson_boltzmann" 设为 1，并将 "-pb_executable" 指向 DelPhi 的路径（本教程中文件名为 'mmpbsa_delphi.vmd'），再次运行以下命令时，程序将调用 DelPhi 计算 PB 能量：

```
#> molaical.exe -mmpbgsa -f mmpbsa_delphi.vmd
```

- 其中，文件 'mmpbsa_apbs_create_file.vmd' 或 'mmpbsa_delphi_create_file.vmd' 比 'mmpbsa_apbs.vmd' 或 'mmpbsa_delphi.vmd' 多出以下参数：
 - "-namd_config_file" input.namd \".
 - "-pb_gb_file" input.apbs \" 或 "-pb_gb_file" input.delphi \"
- "-complex_selection"、"-receptor_selection" 和 "-ligand_selection" 用于选择复合物、受体和配体的分子部分，功能类似 VMD Tk 控制台中的 "atomselect" 命令。
- "-sa_input_apbs" 用于自定义 SASA 计算的 APBS 输入文件。用户可参考材料文件中的模板 "input_sa.apbs"，只需在输入文件（如 'mmpbsa_apbs.vmd' 或 'mmpbsa_delphi.vmd'）中添加一行："-sa_input_apbs" input_sa.apbs \"

通过此机制，用户可以在 "input.namd" 文件中自定义 NAMD 的输入参数，在 "input.apbs" 文件中自定义 APBS 的输入参数，或在 "input.delphi" 文件中自定义 DelPhi 的输入参数。若运行以下命令：

```
#> molaical.exe -mmpbgsa -f mmpbsa_apbs_create_file.vmd
or
#> molaical.exe -mmpbgsa -f mmpbsa_delphi_create_file.vmd
```

且自定义文件（如："input.namd" 或 "input.apbs"）中的参数与 MolAICal 内置参数一致，则将得到相同结果。

- 如上表结果所示，极性贡献为 Pol = Elec + PB，非极性贡献为 Npol = Vdw + SA。

'mmpbsa_apbs_create_file.vmd'、'mmpbsa_delphi_create_file.vmd'、'mmpbsa_apbs.vmd' 或 'mmpbsa_delphi.vmd' 的详细参数列表如下:

usage: mmpbgsa	
-f <arg> mmpbgsa	计算的输入文件路径。
-h	打印帮助信息。
-i mmpbgsa	程序的命令参数。该参数包含外部程序的所有命令行参数，但“-i”参数必须是最后指定的参数，而所有其他标识符（如“-f”等）必须在“-i”之前指定。
-mmpbgsa	执行 MM/PB/GBSA 计算。
-n <arg>	待执行程序 VMD 的名称，默认值为“vmd”。
-o <arg>	是否打印程序的输出内容。
-p <arg>	程序 VMD 的路径。可通过 MolAIcal 环境配置命令进行一次性配置，后续无需重复输入繁琐的路径。
使用方式 (Usage): -mmgbpmbsa -f xxx.vmd -i -system_topology 拓扑文件 (top_file) -system_coords 坐标文件 (coords_file) -trajectory_files 轨迹文件 (filename) [- 参数 (args) ...]	
必选参数 (Mandatory arguments):	
-system_topology	<拓扑文件名 (topology filename) >
-trajectory_files	<轨迹文件名 (trajectory filename) >
-system_coords	<坐标文件名 (coordinates filename) >
可选参数 (Optional arguments):	
-namd_config_file	<NAMD 自定义配置文件 (namd custom configuration file) >
-pb_gb_file	<PB 或 GB 自定义配置文件 (PB or GB custom configuration file) >
-pb_gb_temperature	<温度 (temperature) > -- 默认值: 310K
-topology_type	<拓扑文件类型 (topology filetype) > -- 默认值: auto (自动)
-trj_type	<轨迹文件类型 (trajectory filetype) > -- 默认值: auto (自动)
-force_parameters	<力场参数 (force field parameters) > -- 默认值: auto-detect (自动检测)
-output_filename	<输出文件名 (output filename) > -- 默认值: mmgbpmbsa.log
-debug	<调试级别 (debug level) > -- 默认值: 0。可选值为 0、1、2。若选择“2”，将保存所有生成的临时文件；若选择“1”，将保存部分生成的临时文件；若选择“0”，仅保存结果文件。
-first_frame	<起始帧 (first frame) > -- 默认值: 0
-last_frame	<结束帧 (last frame) > -- 默认值: -1
-frame_stride	<步长 (stride) > -- 默认值: 1
-complex_selection	<复合物选择 (complex selection) > -- 默认值: 空字符串 ("")
-receptor_selection	<受体选择 (receptor selection) > -- 默认值: 空字符串 ("")
-ligand_selection	<配体选择 (ligand selection) > -- 默认值: 空字符串 ("")
-molecular_mechanics	<是否进行气相计算 (do gas-phase calc) > -- 默认值: 0
-namd_executable	<NAMD 的路径 (path to NAMD) > -- 默认值: "namd2"
-namd_cutoff	<分子动力学 (MD) 模拟的截断值 (cutoff for MD simulations) > -- 默认值: 16.0
-namd_switching	<"on" 表示平滑切换函数 (smooth switching function) 在截断距离 (cutoff distance) 处截断范德华 (van der Waals) 势能；"off" 表示关闭平滑切换函数 > -- 默认值: on

-namd_switchdist	<激活切换函数的距离 (distance at which to activate switching function) >-- 默认值: "namd2"
-mm_dielectric	<介电常数 (dielectric constant) >-- 默认值: 1.0
-poisson_boltzmann	<是否进行 PB 计算 (do PB calculation) >-- 默认值: 2。可选值为 0、1、 2。若选择 “0”，将计算 MM/GBSA；若选择 “1”，将使用 Delphi 程序计 算 MM/PBSA；若选择 “2”，将使用 APBS 程序计算 MM/PBSA。
-pb_executable	<DelPhi/APBS 的路径 (path to DelPhi/APBS) >-- 默认值: "apbs"
-pb_radii_type	<PB 半径类型 (type of PB radii) >-- 默认值: bondi
-pb_boundary_flag	<边界条件 (boundary condition) >-- 默认值: sdh
-pb_charge_method	<电荷方法 (charge method) >-- 默认值: spl0
-ion_charge_apbs_negtive	<总阴离子电荷 (total anionic charge) >-- 默认值: -1.0
-ion_charge_apbs_positive	<总阳离子电荷 (total cationic charge) >-- 默认值: 1.0
-ion_apbs_radius	<移动离子种类半径 (mobile ion species radius) >-- 默认值: spl0
-generalized_born	<是否进行 GB 计算 (do GB calculation) >-- 默认值: 0
-gb_exterior_dielectric	<外部介电常数 (external dielectric) >-- 默认值: 78.5
-pb_gb_ion_concentration	<离子浓度 (ion concentration) >-- 默认值: 0.15
-gb_surface_area	<是否进行 LCPO 计算 (do LCPO calculation) >-- 默认值: 0
-gb_surface_gamma	<表面张力 (surface tension) >-- 默认值: 0.0072
-surface_accessibility	<是否进行 SA 计算 (do SA calculation) >-- 默认值: 1，可选值为 1 或 2。若参数为'1'，则通过 VMD 软件计算溶剂可及表面积 (SASA)；若参数 为'2'，则通过 APBS 软件执行计算溶剂可及表面积 (SASA)。
-sa_input_apbs	<用于 SA 计算的 APBS 输入文件 (APBS input file for SA calculation) > -- 默认值: 空字符串 ("")
-sa_radii_type	<SA 半径类型 (type of SA radii) >-- 默认值: bondi
-sa_gamma	<表面张力 (surface tension) >-- 默认值: 0.0072
-sa_beta	<表面偏移量 (surface offset) >-- 默认值: 0.0
-sa_probe_radius	<探针半径 (radius of probe) >-- 默认值: 1.4
-sa_displacement_apbs	<基于有限差分的力计算 (表面积导数) 的位移参数，单位: 埃 (Å) > -- 默认值: 0.2
-sa_sample_points	<采样点数 (number of samples) >-- 默认值: 500
-namd_seed	<用于结果可重复性的 NAMD 种子 (namd seed for reproducibility) >-- 默 认值: 12345
-num_calc_cores	<用于计算的 CPU 核心数 (number of CPU cores for calculations) > -- 默 认值: 1

第三部分：残基能量分解

3.2. 残基能量分解

可以参考上面的方法进行残基能量分解，比如：根据报道，SARS-CoV-2 Mpro 的残基 M165

和配体 N3 有相互作用，本教程就以残基 M165 为例，计算残基 M165 的自由能贡献值。首先，切换到教程目录：

```
#> cd 020-mmpbsa\Decompose
```

残基能量分解的计算脚本和方法与上述示例完全相同。唯一区别在于残基能量分解和配体分子的精确选择，这需要修改以下参数：

```
-complex_selection      "protein and resid 165 or resname LIG" \
-receptor_selection     "protein and resid 165" \
-ligand_selection        "resname LIG" \
```

注："-complex_selection"、"-receptor_selection"和"-ligand_selection"参数用于选择复合物、受体和配体的分子部分，其功能类似于 VMD Tk 控制台中的"atomselect"命令。

运行以下命令可基于 MM/PBSA 快速计算残基自由能贡献：

```
#> molaical.exe -mmpbgsa -f dec_mmpbsa_apbs.vmd
```

输出结果包含每个残基的 ΔG 结合自由能，如下所示：

Delta:				
BOND:	25	-0.0000	0.0000	0.0001
ANGLE:	25	-0.0000	0.0000	0.0000
DIHED:	25	0.0000	0.0000	0.0000
IMPRP:	25	0.0000	0.0000	0.0000
KINETIC:	25	7.2977	0.0000	0.0000
Elec:	25	-2.2492	0.1908	0.9541
Vdw:	25	-5.3395	0.2359	1.1793
PB:	25	1.6785	1.9886	9.9430
SA:	25	-1.2774	0.0397	0.1984
Gas:	25	-7.5887	0.3867	1.9333
Sol:	25	0.4011	1.9851	9.9256
Pol:	25	-0.5707	1.9595	9.7977
Npol:	25	-6.6169	0.2676	1.3378
G binding:	25	-7.1876	+/- 1.9791	9.8957

*所有能量值的单位均为千卡/摩尔 (kcal/mol)

注：

- ✧ 用户可采用上述类似方法，通过 MolAICal 计算残基对(pairwise)和单残基(per-residue)的自由能贡献。不同操作系统或 NAMD 版本可能导致结果存在差异，但若保留两位小数，结果将完全相同或高度相近。
- ✧ "输入文件与结果分析"的操作方式与上述结果高度相似，请参考前文说明。

总体而言，若参数预设得当，MolAICal 提供了一种快捷的 MM/PBSA 计算方法——仅需输入一条命令即可完成计算。

第四部分：熵的计算

3.3. 通过 Carma 和 MolAICal 进行熵的计算

3.3.1. 下载 Carma

Carma 可以在 Windows 或 Linux 操作系统上运行。从 <https://github.com/glykos/carma>, <https://github.com/glykos/carma>, <http://utopia.duth.gr/~glykos/progs>, 或 <http://utopia.duth.gr/~glykos> 上下载新版本 Carma。

推荐使用最新版本的 Carma 进行熵的计算，之前的版本会出现“NaN”的错误等，具体请查看下面的链接：

<https://groups.google.com/g/carma-molecular-dynamics/c/KpyY5sEkrj4>

本教程选取 100 帧轨迹进行熵的计算。用户可以根据自己的实际情况选择合适的帧数，越多帧数越耗费计算时间。

根据 Carma 的许可证 (<https://github.com/glykos/carma/blob/master/LICENSE>)，该许可证未限制使用、复制、修改、合并、发布、分发等权利。因此，MolAICal 可直接集成并调用 Carma 来拟合分子并计算分子熵，而无需用户再次下载 Carma。

选项：此处诸如'mpro.dcd'和'mpro.psf'等输入文件包含蛋白质、配体、离子、水分子等组分，可能占用大量计算机内存。熵值计算可能耗费较多时间和内存。为节省内存并提升计算速度，用户可通过以下命令分别提取复合物（受体与配体）、受体及配体的轨迹：

1. 复合物（第 1 分子）：

```
#> molaical.exe -call run -n vmd -c stripDCD -i -dispdev text -psf "mpro.psf" -args protein,or,resname,LIG  
"mpro.dcd" "complex" mpro.psf mpro.pdb
```

注："complex"为输出文件前缀，程序将参照 mpro.psf 和 mpro.pdb 生成前缀为"complex"的文件，最终输出：'complex.psf', 'complex.pdb'和'complex.dcd'。

2. 受体（第 2 分子）：

```
#> molaical.exe -call run -n vmd -c stripDCD -i -dispdev text -psf "mpro.psf" -args protein "mpro.dcd" "protein"  
mpro.psf mpro.pdb
```

注："protein"为输出文件前缀，程序将参照原文件生成：'protein.psf', 'protein.pdb' 和 'protein.dcd'。

3. 配体（第 3 分子）：

```
#> molaical.exe -call run -n vmd -c stripdcd -i -dispdev text -psf "mpro.psf" -args resname,LIG "mpro.dcd"  
"ligand" mpro.psf mpro.pdb
```

注："ligand"为输出文件前缀，生成文件包括：'ligand.psf', 'ligand.pdb', 和 'ligand.dcd'。

3.3.2. 计算复合物的熵

```
#> cd 020-mmmpbsa\entropy
```

切换到文件夹“com”。这一步是去掉复合物的旋转和平移：

```
#> molaical.exe -call run -c entropy -i -v -fit -force -atmid ALLID -segid A -segid C complex.dcd  
complex.psf
```

注：生成的文件"carma.fitted.dcd"采用固定命名，且该文件仅包含"-segid"参数指定的对应内容。用户可依据此提示优化调整命令参数以满足研究需求。

-call:	当值为"run"时，将调用外部程序。
-c:	计算方式。当值为"entropy"时，将通过 Carma 计算熵值。
-i:	在"-i"后输入 Carma 相关命令。该命令行参数"-i"必须放置在其他参数(如"-c"参数)之后。
-v:	详细信息输出
-fit:	从 DCD 帧中移除全局旋转和平移
-force:	指示 Carma 在熵值变为未定义前停止计算。
-atmid:	基于原子名称的原子选择
-segid:	基于片段标识符的原子选择(仅限 dcd-psf 文件)

这一步是计算熵值：

```
#> molaical.exe -call run -c entropy -i -v -cov -eigen -mass -force -temp 310 -atmid ALLID -segid  
A -segid C -last 25 carma.fitted.dcd complex.psf >& com.log &
```

-call:	当值为"run"时，将调用外部程序。
-c:	计算方式。当值为"entropy"时，将通过 Carma 计算熵值。
-i:	在"-i"后输入 Carma 相关命令。该命令行参数"-i"必须放置在其他参数(如"-c"参数)之后。
-v:	详细信息输出
-cov:	计算方差-协方差矩阵
-eigen:	计算协方差矩阵的特征向量和特征值
-mass:	对协方差矩阵使用质量加权
-force:	指示 Carma 在熵值变为未定义前停止计算。
-atmid:	基于原子名称的原子选择
-segid:	基于片段标识符的原子选择(仅限 dcd-psf 文件)

3.3.3. 计算受体的熵

切换到文件夹“rec”。这一步是去掉受体的旋转和平移：

```
#> molaical.exe -call run -c entropy -i -v -fit -atmid ALLID -segid A protein.dcd protein.psf
```

这一步是计算熵值：

```
#> molaical.exe -call run -c entropy -i -v -cov -eigen -mass -force -temp 310 -atmid ALLID -segid  
A -last 25 carma.fitted.dcd protein.psf >& rec.log &
```

3.3.4. 计算配体的熵

切换到文件夹“lig”。这一步是去掉配体的旋转和平移：

```
#> molaical.exe -call run -c entropy -i -v -fit -force -atmid ALLID -segid C ligand.dcd ligand.psf
```

这一步是计算熵值:

```
#> molaical.exe -call run -c entropy -i -v -cov -eigen -mass -force -temp 310 -atmid ALLID -segid C -last 25 carma.fitted.dcd ligand.psf>& lig.log &
```

3.3.5. 计算熵值

请检查 log 文件: “com.log”, “rec.log” 和 “lig.log”, 并在文件“com.log”, “rec.log” 和 “lig.log” 中找到 Andricioaei 或 Schlitter 的熵值。例如: 可以在“com.log”文件中找到 Andricioaei 或 Schlitter 的熵值 (见下图):

```
Writing postscript file carma.fitted.dcd.varcov.ps.
Calculation of eigenvectors and eigenvalues ...
Asking for optimal workspace size : 487662
Starting the calculation ...
Done. Now sorting ...

Entropy calculation will ignore negative eigenvalues !

Entropy (Andricioaei) using only 7220 eigenvalues is 15522.260938 (J/molK)
Entropy (Schlitter) using only 12357 eigenvalues is 10225.524850 (J/molK)
done.
All done in 63.4 minutes.
```

切换到上一层目录，基于日志文件运行 MolAICal 的操作方法，需使用"com.log"、"rec.log" 和"lig.log"文件，执行步骤如下:

```
#> molaical.exe -entropy -if -c com/com.log -r rec/rec.log -l lig/lig.log -t 310.0
```

这个结果显示:

Entropy Type: Andricioaei

The entropy (T * delta S) = -25.3034 (kcal/mol)

Entropy Type: Schlitter

The entropy (T * delta S) = -32.3195 (kcal/mol)

各参数定义如下:

- entropy:** 表示进行熵变计算 (ΔS)
- c:** 复合物熵值或其结果文件 (此处应为结果文件)
- r:** 受体 (或第一分子) 熵值或其结果文件 (此处应为结果文件)
- l:** 配体 (或第二分子) 熵值或其结果文件 (此处应为结果文件)
- t:** 分子动力学模拟的开尔文温度
- if:** 若输入参数包含"-if"，则从结果文件中读取熵值

注意: Windows 下计算熵值速度很慢，可能只有 Schlitter 的熵值；推荐使用 linux 系统进行熵的计算。推荐在 Linux 系统下计算熵值。

如果考虑熵值，MM/PBSA 通过以下方程计算：

$$\Delta G_{\text{bind}} = \Delta H - T\Delta S \approx \Delta E_{\text{MM}} + \Delta G_{\text{sol}} - T\Delta S$$

$$\Delta E_{\text{MM}} = \Delta E_{\text{internal}} + \Delta E_{\text{ele}} + \Delta E_{\text{vdw}}$$

$$\Delta G_{\text{sol}} = \Delta G_{\text{PB}} + \Delta G_{\text{SA}}$$

$$\Delta G_{\text{SA}} = \gamma * \text{SASA} + \beta$$

正如上文所述，熵能值未被纳入考虑，具体如下：

delta G binding: -66.5006 +/- 2.7810 (kcal/mol)

使用 Andricioaei 熵

$$\Delta G_{\text{bind}} = -66.5006 - (-25.3034) = -41.1972 \text{ (kcal/mol)}$$

使用 Schlitter 熵

$$\Delta G_{\text{bind}} = -66.5006 - (-32.3195) = -34.1811 \text{ (kcal/mol)}$$

注：若分子动力学 (MD) 模拟过程中未发生结合诱导的结构变化，或不同系统间的结构变化高度相似，则可省略熵计算，此时仅使用不考虑熵值的结合自由能 (ΔG) 数值就足够了。

选项：

◆ MolAICal 同样提供了一种简便的熵计算方法，如下所示：

```
#> molaical.exe -call run -c script -n runscript -i com rec lig
```

位于 '-i' 之后的字符串是将被空格分割的参数列表。分割后得到的字符串将依次替换脚本文件中的占位符 \$molargs1, \$molargs2, \$molargs3 等。脚本中占位符 \$molargs1, \$molargs2, \$molargs3 等的数量，必须与 '-i' 之后字符串经分割后得到的参数数量相同。其中，参数 '-n' 用于指定脚本文件名称，此处命名为 "runscript"，如：

```
# 1. calculate complex part (rename in win: move /y carma.fitted.dcd complex_fit.dcd)
cd $molargs1
molaical.exe -call run -c entropy -i -v -fit -force -atmid ALLID -segid A -segid C complex.dcd complex.psf
molaical.exe -call run -c entropy -i -v -cov -eigen -mass -force -temp 310 -atmid ALLID -segid A -segid C -last 25 carma.fitted.dcd
complex.psf > com.log
cd ..

# 2. calculate receptor part
cd $molargs2
molaical.exe -call run -c entropy -i -v -fit -atmid ALLID -segid A protein.dcd protein.psf
molaical.exe -call run -c entropy -i -v -cov -eigen -mass -force -temp 310 -atmid ALLID -segid A -last 25 carma.fitted.dcd protein.psf >
rec.log
```

```
cd ..  
  
# 3. calculate receptor part  
cd $molargs3  
molaical.exe -call run -c entropy -i -v -fit -force -atmid ALLID -segid C ligand.dcd ligand.psf  
molaical.exe -call run -c entropy -i -v -cov -eigen -mass -force -temp 310 -atmid ALLID -segid C -last 25 carma.fitted.dcd ligand.psf>  
lig.log  
cd ..  
  
# in windows: | findstr /I /R "entropy" > results.log    in linux: | grep -i "entropy" > results.log  
molaical.exe -entropy -if -c $molargs1/com.log -r $molargs2/rec.log -l $molargs3/lig.log -t 310.0 > results.log
```

注意: "molaical.exe" 字符本身无需修改, MolAICal 将自动替换命令中的 "molaical.exe" 为
其实际路径。

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.ta  
r
```

- ✧ 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
```