Epock

安装

我们要求的编译包在 Mac OS 和 Linux 下可做，下载地方如下：

1.预编译包

预编译包包括 Epock’s 编译，VMD 的 Tcl 插件，Python 脚本用于 Epock 输出。

结构如下：

```
epock-1.0.0rc-Darwin-i386/
├── INSTALL
├── README
├── bin
│   └── epock
└── share
    └── epock
        ├── plugins
        │   └── vmd
        │       └── epock1.0
        │           ├── epock.tcl
        │           ├── epock_gui.tcl
        │           ├── epock_results_dynamic_window.tcl
        │           ├── epock_results_window.tcl
        │           ├── graph_radius.tcl
        │           ├── pkgIndex.tcl
        │           ├── run_tk.tcl
        │           └── table.tcl
        └── scripts
            ├── plot.py
            ├── plot_contribution.py
            └── plot_profile.py
```

我们建议放置在$PATH 路径下：

tar xf epock-1.0.0rc-Darwin-i386.tar.gz

$ sudo mv epock-1.0.0rc-Darwin-i386/bin/epock /usr/local/bin/

$ sudo mv epock-1.0.0rc-Darwin-i386/share/epock /usr/local/share/

2.2 对于资源包

要求：

CMake version >= 2.8
A compiler that supports C++11 standards (e.g GCC version >=4.7)

编译：

```
$ tar xf epock-1.0.0rc.tar.gz
$ cd epock-1.0.0rc
$ mkdir build
$ cd build
$ cmake ..
$ make
$ sudo make install
```

安装 VMD 插件

复制粘贴 Epock 插件进入正确的目录（查看之前），在$HOME/.vmdrc 文件下添加如下行：

set auto_path [linsert $auto_path 0 "/path/to/directory/share/epock/plugins/vmd"]

vmd_install_extension epock epock_tk "Analysis/Epock"

特别提醒：安装预编译版本。如果你安装的 epock 为预编译版本，目录并非你的$PATH,你可能需要改变 epock 的扩展路径在你的 VMD 插件。This can be done by modifying the value of ::epock::exec_path in epock_gui.tcl (line 47).

快速开始

设置文件

原理

如下是一个简单的设置文件的例子

```
[DEFAULT]
grid_spacing = 0.5
contribution = residue
contiguous = true
contiguous_cutoff = 4.0

[pock1]
include_sphere          =    1.90    26.70   -21.91    8.0
exclude_sphere          =   10.26    25.14   -20.03    6.0
contiguous_seed_sphere =    1.90    26.70   -21.91    4.0
```

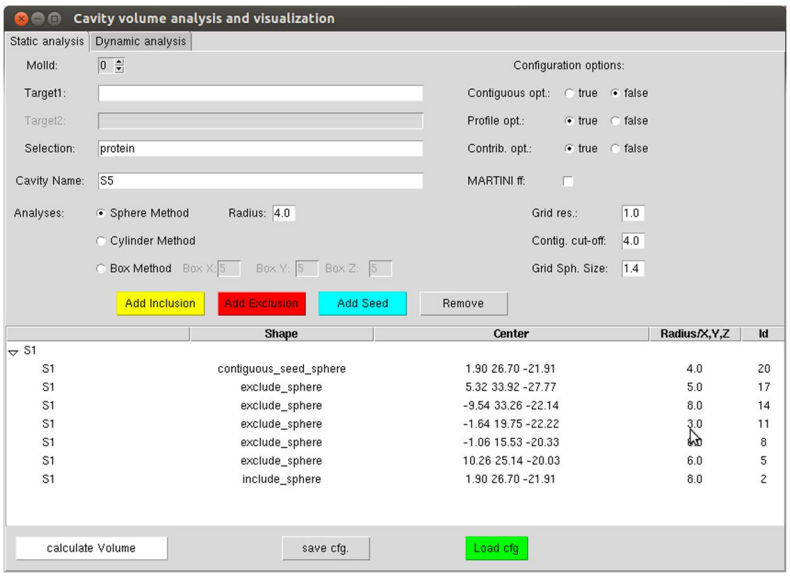
```
[pock2]
include_sphere            =     -0.14       5.56    -20.79     8.0
exclude_sphere            =      3.69      12.71    -19.56     6.0
exclude_sphere            =     10.55      -1.65    -20.54     6.0
contiguous_seed_sphere =     -0.14       5.56    -20.79     4.0
```

定义信息与参数可以参考如下：

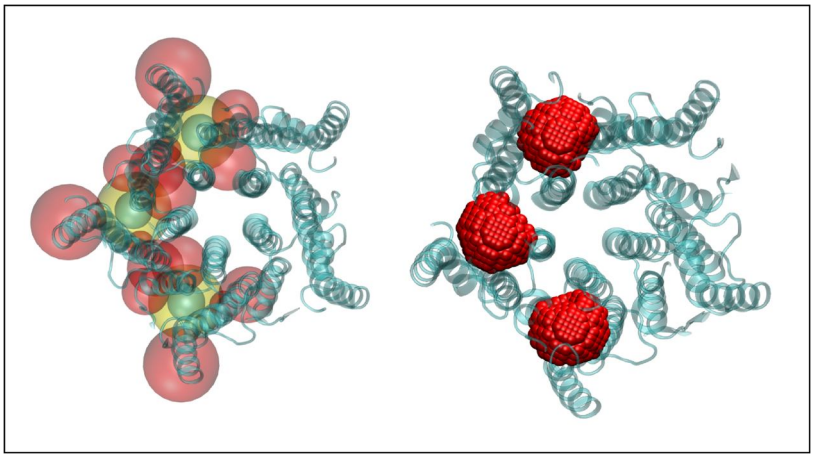
定义口袋最大的 Maximum Englobing Region (MER)，是非常直观的使用坐标对于中心原子的选择。因此设置文件可以手写-但是 VMD 插件可以直观的使用达到次目的。

使用 VMD 插件生成参数文件

在安装 VMD 插件以后，你可以通过 Extension -> Analysis -> Epock.设置



每个腔体定义通过名字（Givem in the “Cavity Name”entry），可以创建包含、排除和种子形状使用“Add inclusion”, “Add Exclusion”, “Add Seed” 按钮。如果位置没有设置的很好，可以使用“Remove”按钮。可以通过在窗口双击隐藏形状。你可以在 VMD OpenGL 窗口查看起始格点位置。



定义 MER 使用 Epock’s VMD 插件。左：每个 MER 联系包括黄色和扩展的红色；设置的位点种子显示蓝绿色。右：结果

保存可以点击：save cfg 按钮。

Epock 轨迹格式和转化

Epock 一个重要作用是能够高效的计算随着 MD 轨迹的体积变化。输入轨迹的格式至关重要。此段主要讲的 NAMD 转 Gromacs，因为用不到，所以不写了

Epcok 正式开搞

使用命令行工具

```
$ ./epock -h
epock v1.0.0rc

Calculates the volume of cavities along a trajectory.

usage: epock [options] -s coor.pdb -c conf.cfg

File I/O:
----------
   -s              Input              Coordinate file (pdb format)
   -c              Input              Configuration file
   -f              Input, Opt         Trajectory file (gromacs xtc format)
   -o              Output, Opt        Volume output file (default: volume.dat)

Options:
--------
   -b                          First frame (ps) to read from trajectory
   -e                          Last frame (ps) to read from trajectory
   --ox                        Output cavity trajectory file (default: no)
   --dt <timestep>             Only use frame when t MOD dt = first time (ps)
   --martini                   Use martini vdW radii i.e. 4.7 for beads and 2.1 for probe
   --mol                       Calculate volume of a molecule instead of a cavity
   --radii <filename>          Use custom atom radii
   --dry-run                   Stop after basic initialization and write the grid as a PDB file
   -h, --help                  Show this message and exit
   --version                   Show the version number and exit
   -v                          Enable verbose mode
```

Epock 强制轨迹文件与拓扑文件

$ epock -s conf.pdb -f traj.xtc -c config.cfg –ox

使用 VMD 插件

在定义了格点以后或者加载了你的设置文件，你可以特别的选择 Selection 构架如果一些原子需要被忽略。默认的，只有蛋白原子会被考虑。

计算 pocket 静态体积，只需点击“calculate Volume”按钮

计算 pocket 加载轨迹后体积，切换至“dynamic analysis”tab。可以设置开始和结束的轨迹 frames 和时间步骤。其也可以 fit 轨迹。最终，点击“计算体积”按钮。

结果分析：

VMD

自由的可视化检查

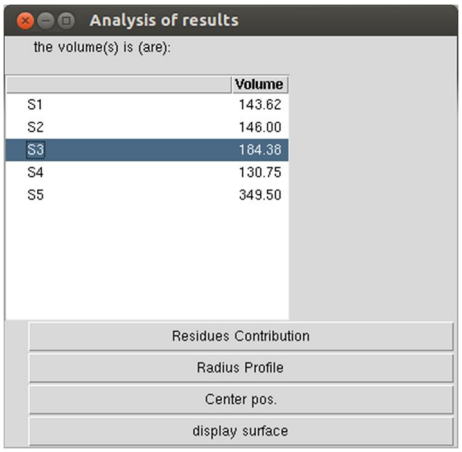
当 Epocket 计算完成第一件事情是查看自由空间是否适合口袋。这个事情非常容易做到。使用 Epock 设置输出自由空间作为一个拓扑和一个轨迹文件（--ox option）。蛋白和自由空间轨迹能够被加载在 VMD 相同时间。

$ vmd prot.pdb prot.xtc -f cav.pdb cav.xtc

一个快速的可视化检测可以非常容易的使用蛋白作为 wireframe surface，自由空间作为 van der Walls spheres。对于自由空间，Epock 使用 van der Walls radius of 1.4A。打开 Tk (Extensions -> Tk Console)输入：

```
% [atomselect 1 "all"] set radius 1.4
```

当运行 Epock 通过 VMD，之后点击 “calculate volume” 按钮在“static analysis”框中，当计算完成会弹出一个新窗口。



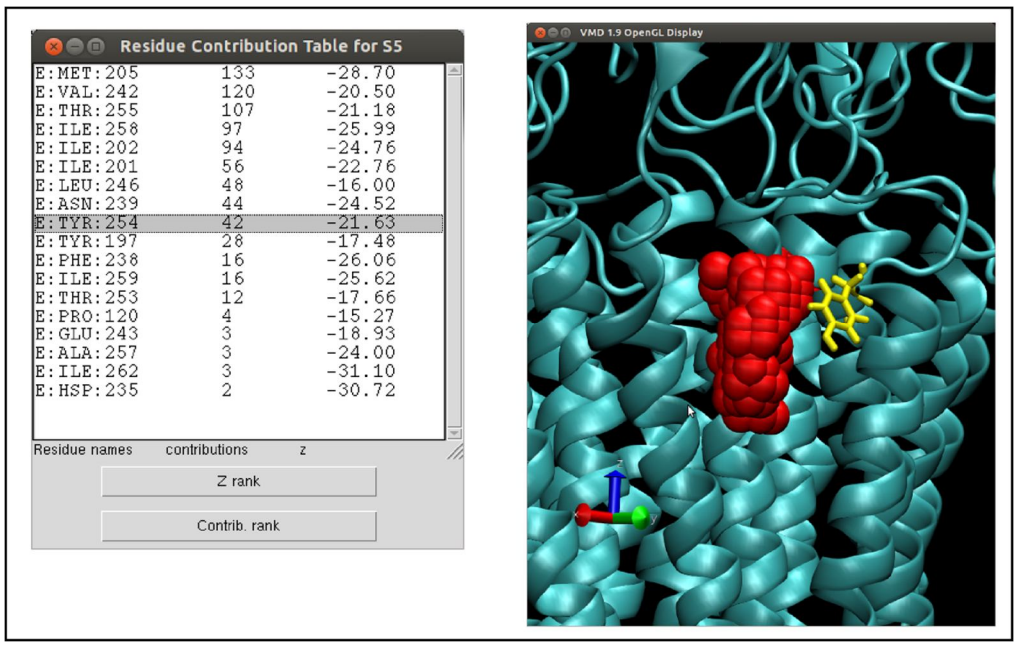
静态结果体积计算图

更多描述可以点击下面的按钮：

“**Residues contribution**”打开一个新的窗口里面展示了残基涉及到的口袋

“**Radius profile**”给一个空隙的轮廓直径

“**Center pos**”加载 pdb 文件计算中心孔径。



残基贡献窗口（左）和 OpenGL 窗口（右），点击残基右侧会显示

动力学结果窗口

“Volume vs Time”顾名思义

“Residue contribution”允许用户绘画特别的残基对于时间

“Radius profile”绘画一个平均半径作为最大和最小的 z 值（对于空隙非常重要）.

“Center pos”和静止分析类似

Python Scripts

Python version 2.7.x, the Python interpreter

matplotlib version >= 1.1.1, the Python plot library

Numpy version >= 1.6.1, a package for scientific computing with Python

Plot.py

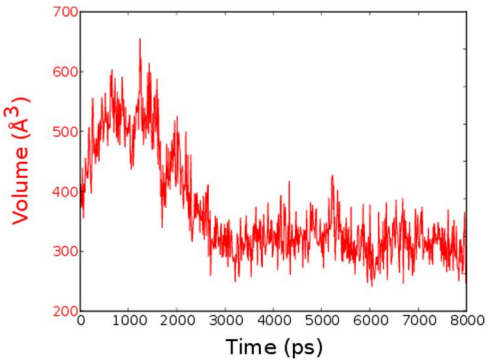
Description

这个脚本绘画一个时间线画图对于每个 pocket 体积，每个文件通过命令行

使用：

```
$ python plot.py -h
usage: plot.py [-h] [-o OUTPUT] filename [filename ...]
This script draws a timeline plot of each pocket volume in each file passed to
command-line.
positional arguments:
  filename                volume data file
optional arguments:
  -h, --help              show this help message and exit
  -o OUTPUT, --output OUTPUT   output file name
```

输出：

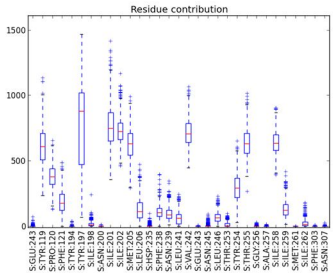


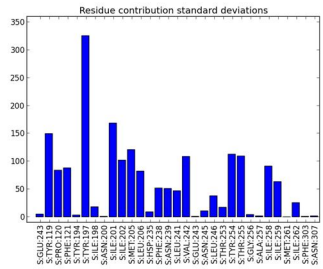
800 个轨迹

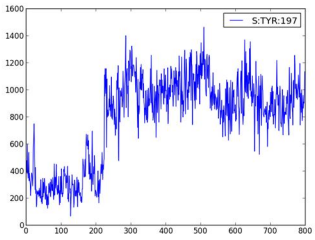
plot_contribution.py

如下：

```
python plot_contribution.py -h
usage: plot_contribution.py [-h] [-o OUTPUT] [-n N] [-s]
                            [-r residue [residue ...]]
                            filename
This script draws a boxplot of each atom contribution to the cavity.
positional arguments:
  filename                contribution data file
optional arguments:
  -h, --help              show this help message and exit
  -o OUTPUT, --output OUTPUT   output file name
  -n N                    show n greatest contributions
  -s, --stdev             only plot standard deviations
  -r residue [residue ...]   plot specific residues along time
```







plot_profile.py

Description

残基贡献

```
Usage
usage: plot_profile.py [-h] [-o OUTPUT] filename [filename ...]
This script plots the minimum, maximum and average profile from a profile file
passed to command-line.
positional arguments:
  filename                contribution data file
optional arguments:
  -h, --help              show this help message and exit
  -o OUTPUT, --output OUTPUT   output file name
```