3DRISMHI User Manual

香港科技大學
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

Version 0.255

Contents

1 Overview

The 3DRISMHI is an open-source software package to calculate the atomic distributions of solvents around given solutes based on integration equation theory (IET) or liquid and hydrophobicity induced density inhomogeneity (HI) theory[1–3]. With the structure of solutes, the force field parameters of solute, the solvent-solvent correlations (provided in the package or generated from MD simulations), the molecular structure of solvents (provided in the package or generated from trajectories) and some experimental parameters (provided in the package), the 3DRISMHI can efficiently calculate the solvent density distrubution.

The 3DRISMHI can use multi cores (of one computer) to speed up the calculation. The memory required to perform a 3DRISMHI calculation is probably huge, so by default the allocated memory will not exceed the physical RAM installed on your computer (you can bypass this check via -ignore-memory-capacity). 3DRISMHI doesn't provide MPI or GPU paralleling due to highly and frequently exchange of memories.

The 3DRISMHI is an independent software that can run on Linux, Mac OS, Windows Subsystem Linux and Windows Cygwin. The only required package is FFTW version 3. More features will be avaiable if compiled with the following packages: libz, GROMACS. The 3DRISMHI doesn't have any graphical interface, and need to run in terminals or computer clusters.

The 3DRISMHI is developed and maintained by Siqin Cao (PhD) from the Hong Kong University of Science and Technology (HKUST), and supported by Xuhui Huang (Professor PhD) from HKUST.

2 Theory

2.1 Force field

The 3DRISMHI adopts the well received molecular force field model to compute the solute-solvent interactions, which consists of a Lennard-Jones (LJ) potential and an electrostatic potential (Coulomb by default):

$$v_{ij}(r_{ij}) = 4\varepsilon_{ij}\left(\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right) + \frac{q_i q_j}{4\pi\varepsilon_0 r_{ij}}$$

In the LJ potential, the mixing rule of ε is

$$\varepsilon_{ij} = \sqrt{\varepsilon_i \varepsilon_j}$$

Two mixing rules of ε are provided:

$$\begin{aligned}\text{arithmetic}: \sigma_{ij} &= \frac{\sigma_i + \sigma_j}{2}\\ \text{geometric}: \ \sigma_{ij} &= \sqrt{\sigma_i \sigma_j}\end{aligned}$$

Several electrostatic potentials are provided (for RISM and HI only; not for reporting solute-solvent electrostatic interaction):

$$\begin{aligned}\text{Coulomb}: & v_{ij}^{\text{ES}}(r_{ij}) = \frac{q_i q_j}{4\pi\varepsilon_0 r_{ij}}\\ \text{Dielectric}: & v_{ij}^{\text{ES}}(r_{ij}) = \frac{q_i q_j}{4\pi\varepsilon_0\varepsilon_r r_{ij}}\\ \text{Yukawa}: \quad & v_{ij}^{\text{Yukawa}}(r_{ij}) = \frac{q_i q_j}{4\pi\varepsilon_0}\frac{e^{-\kappa r_{ij}}}{r_{ij}}\end{aligned}$$

where ε_r is the dielectric constant, and $\kappa = 1/r_{\text{Yukawa}}$ is the Debey wavenumber (or overridden by a user defined number).

The Particle Mesh Ewald (PME) method is adopted to calculate the Coulomb potential:

$$\frac{1}{r} = \frac{\mathrm{erf}(\gamma r)}{r} + \frac{\mathrm{erfc}(\gamma r)}{r}$$

where $\gamma = 2/r_C$, and r_C is the cutoff distance for short range part of Coulomb interactions. The long range part of the Coulomb potential $\mathrm{erfc}(\gamma r)/r$ and the Yukawa potential are both calculated with Fourier transformations.

2.2 Equation one of IET: the Ornstein-Zernike equation

The OZ equation:

$$\begin{aligned} h_{ij}(\vec{r}_i, \vec{r}_j) &= c_{ij}(\vec{r}_i, \vec{r}_j) + \int c_{ik}(\vec{r}_i, \vec{r}_k) h_{kj}(\vec{r}_k, \vec{r}_j) \rho_k^b d^3\vec{r}_k \\ &\equiv c_{ij}(\vec{r}_i, \vec{r}_j) + c_{ik}(\vec{r}_i, \vec{r}_k) *_k h_{kj}(\vec{r}_k, \vec{r}_j) \end{aligned} \tag{1}$$

The 3DSSOZ or 3DRISM equation:

$$h_{ij} = c_{ij} + c_{ik} *_k \hbar_{kj} \tag{2}$$

where the total solvent-solvent correlation function consists of the inter-molecular correlation h_{kj} and the intra-molecular correlation[3] $\delta_{kj}(r_{kj} - b_{kj})$:

$$\hbar_{kj} = h_{kj}^{\mathrm{inter}} + h_{kj}^{\mathrm{intra}} = h_{kj} + \delta_{kj}(r_{kj} - b_{kj}) \tag{3}$$

In most papers of 3DRISM, the susceptibility (χ)[4]:

$$1 + \hbar_{kj} = 1 + h_{kj}^{\mathrm{inter}} + h_{kj}^{\mathrm{inner}} = \omega_{kj} + h_{kj} = \chi_{kj} \tag{4}$$

Finally, the 3DRISM equation with the renormalization of Coulomb:

$$\begin{aligned} h_{ij} &= c_{ij}^{\mathrm{R}} + c_{ik}^{\mathrm{LR}} *_k \hbar_{kj} + h_{ij}^{\mathrm{LR}} \\ h_{ij}^{\mathrm{LR}} &= c_{ij}^{\mathrm{LR}} + c_{ik}^{\mathrm{LR}} *_k \hbar_{kj} \\ c_{ij}^{\mathrm{LR}} &= -v_{ij}^{\mathrm{LR}} \end{aligned} \tag{5}$$

2.3 Equation two of IET: the closure equation

The density functional form of the closure equation is:

$$g_{ij} = \exp\left[-v_{ij} + \int c_{ik} *_k \frac{\partial \hbar_{kj}(\lambda)}{\partial \lambda} d\lambda\right] \tag{6}$$

By expanding the integration, the general form of the closure equation can be derived:

$$g_{ij} = \exp\left[-v_{ij} + \int c_{ik} *_k \hbar_{kj} + B_{ij}\right] = \exp\left[-v_{ij} + h_{ij} - c_{ij} + B_{ij}\right] \tag{7}$$

where B_{ij} is called the bridge function.

This software support the following closures[3,4]. Please choose the closure(s) that suitable for your systems.

2.3.1 HNC/MSA based closures

The Hyper-Netted-Chain (HNC) closure:

$$g_{ij} = \exp\left[-v_{ij} + h_{ij} - c_{ij}\right] \tag{8}$$

The Mean-Spherical-Approximation (MSA):

$$\begin{aligned} v_{ij} < \infty &: c_{ij} = -v_{ij} \\ v_{ij} \to \infty &: h_{ij} = -1 \end{aligned} \tag{9}$$

The Kobryn-Gusarov-Kovalenko (KGK) closure:

$$g_{ij} = \max\left\{0, 1 - v_{ij} + h_{ij} - c_{ij}\right\} \tag{10}$$

The Partial-Linear-HNC (PLHNC) closure:

$$g_{ij} = \begin{cases} \exp\left[-v_{ij} + h_{ij} - c_{ij}\right], & \ln g_{ij} \le C \\ -v_{ij} + h_{ij} - c_{ij} - C + e^{C}, & \ln g_{ij} \ge C \end{cases} \tag{11}$$

The Kovalenko-Hirata (KH) closure:

$$g_{ij} = \begin{cases} \exp\left[-v_{ij} + h_{ij} - c_{ij}\right], & g_{ij} \le 1 \\ 1 - v_{ij} + h_{ij} - c_{ij}, & g_{ij} \ge 1 \end{cases} \tag{12}$$

The PSE-n expansion of the HNC closure:

$$g_{ij} = \begin{cases} \exp\left[-v_{ij} + h_{ij} - c_{ij}\right], & g_{ij} \le 1 \\ 1 + \sum_{m=1}^{n} \frac{1}{n!}(-v_{ij} + h_{ij} - c_{ij})^{n}, & g_{ij} \ge 1 \end{cases} \tag{13}$$

(PSE1 is equivalent to KH. This software support PSE1 to PSE10)

2.3.2 Bridge function based closures

The Percus-Yevick (PY) closure:

$$g_{ij} = e^{-v_{ij}}\left(1 + h_{ij} - c_{ij}\right) \tag{14}$$

The HNC-Bridge (HNCB) closure:

$$g_{ij} = \exp\left[-v_{ij} + h_{ij} - c_{ij} - \frac{1}{2}\left(h_{ik} *_{k} \hbar_{kj}\right)^{2}\right] \tag{15}$$

The D2 (D2 with **-cmd ssoz**) closure[1,3]:

$$g_{ij} = \exp\left[-v_{ij} + h_{ij} - c_{ij} - \frac{1}{2}\left(c_{ik} *_{k} \hbar_{kj}\right)^{2}\right] \tag{16}$$

The D2-MSA (D2 with **-cmd rism**) closure[2,3]:

$$g_{ij} = \exp\left[-v_{ij}^{\mathrm{SR}} + h_{ij} - c_{ij}^{\mathrm{R}} - \frac{(h_{ij} - c_{ij}^{\mathrm{R}})^{2}}{2}\right] \tag{17}$$

The Martynov-Sarkisov (MS) closure:

$$g_{ij} = \exp\left[-v_{ij} + \sqrt{1 + 2(h_{ij} - c_{ij})} - 1\right] \tag{18}$$

In order to avoid the numerical crash at $h_{ij} - c_{ij} < -0.5$, the MSHNC (MSHNC) closure is also implemented in 3DRISMHI:

$$g_{ij} = \begin{cases} \exp\left[-v_{ij} + h_{ij} - c_{ij}\right], & h_{ij} - c_{ij} \geq 0 \\ \exp\left[-v_{ij} + \sqrt{1 + 2(h_{ij} - c_{ij})} - 1\right], & h_{ij} - c_{ij} \leq 0 \end{cases} \tag{19}$$

The BPGG closure:

$$g_{ij} = \exp\left[-v_{ij} + (1 + \alpha(h_{ij} - c_{ij}))^{1/\alpha} - 1\right] \tag{20}$$

In order to avoid the numerical crash at $h_{ij} - c_{ij} < -1/\alpha$, the BPGGHNC (BPGGHNC) closure is implemented in 3DRISMHI:

$$g_{ij} = \begin{cases} \exp\left[-v_{ij} + h_{ij} - c_{ij}\right], & h_{ij} - c_{ij} \geq 0 \\ \exp\left[-v_{ij} + (1 + \alpha(h_{ij} - c_{ij}))^{1/\alpha} - 1\right], & h_{ij} - c_{ij} \leq 0 \end{cases} \tag{21}$$

The Modified Verlet bridge (VM):

$$g_{ij} = \exp\left[-v_{ij} + h_{ij} - c_{ij} - \frac{1}{2}\frac{(h_{ij} - c_{ij})^2}{1 + \alpha \times (h_{ij} - c_{ij})}\right] \tag{22}$$

where α is a constant. In particular, the modified HNC (MHNC) closure takes $\alpha = 0.8$.

The MP closure:

$$g_{ij} = e^{-v_{ij}}\left[(1 + \alpha)e^{(h_{ij} - c_{ij})/(1+\alpha)} - \alpha\right] \tag{23}$$

2.3.3 User defined closures

(For developlers only:) This software also provids three user defined closures: "user1", "user2" and "user3". The name of these three closures can be changed in line 197-199 of `rismhi3d.cpp`, and the code to perform these closures are in line 230-244 of `main-rism.cpp`. The default code of these closures are doing HNC closure, and please change them with your own code if you want to add more closure equations.

The closures like HNCB requires additional convolutions of $h_{ik} *_k \hbar_{kj}$. This kind of additional convolution is performed in the function prepare_closure() of `main-rism.cpp`. Please follow the code of HNCB to include additional terms in your new closures.

2.4 Equation three of 3DRISMHI: the HI equation

In the hydrophobicity induced density inhomogeneity (HI) theory, the global average density constants ρ_γ^b is substituted with spacial inhomogeneous local average densities $\rho_\gamma^S(\vec{r}_\gamma) = \rho_\gamma^b n_\gamma(\vec{r}_\gamma)$. This density is slowly varying and is called the slowly varying component of the liquid density in the LCW theory, or the dimensionless density in the HI theory. The dimensionless density is calculated with the HI equation:

$$\ln n_\gamma - \ln \lambda = \zeta_{\gamma\mu} *_\mu n_\mu \Theta(\rho_\mu - 0^+) + ABe^{(1-1/n_\gamma)/A} \tag{24}$$

The final density distribution of liquid in the HI theory is:

$$\rho_\gamma(\vec{r}_\gamma) = \rho_\gamma^b n_\gamma(\vec{r}_\gamma) g_\gamma(\vec{r}_\gamma) \tag{25}$$

2.5 Self-consistent field iterations

Both the RISM equation and the HI equation are solved in a self-consistent-field (SCF) manner. The SCF iterations of HI is simply done in the following way:

$$\ln n_\gamma^{(n+1)} = \ln \lambda + \zeta_{\lambda\mu} *_\mu n_\mu^{(n+1)} \Theta(\rho_\mu - 0^+) + ABe^{(1-1/n_\gamma^{(n+1)})/A} \tag{26}$$

while the SCF iterations of RISM is performed as follows:

$$\begin{aligned} g^{(n+1)} &= \exp\{-v + c^{(n)} * \hbar + B^{(n)}\} \\ c^{(n+1)} &= g^{(n+1)} - 1 - c^{(n)} * \hbar \end{aligned} \tag{27}$$

In both SCF iterations of RISM and HI, the Direct Inversion in the Iterative Subspace (DIIS) method[5–7] can be used to celerate and stabilize the convergence of SCF iterations. For a self-consistent equation $\mathbf{f} = g(\mathbf{f})$, the DIIS method adopts a linear combination of the approximation errors from the previous iterations to predict the next step,

$$\mathbf{f}_{n+1} = \mathbf{f}_n + \sum_i (\mathbf{f}_i - g(\mathbf{f}_i))c_i \tag{28}$$

where the coefficients c_i are calculated from the minimization of the historical error,

$$L = \sum_i \left| \mathbf{f}_i - g(\mathbf{f}_i) \right|^2 + \lambda \left(\sum_i c_i - 1 \right) \tag{29}$$

3 Compilation guidance

3.1 Binaries in the package

There are totally four binary files after compilation:

- `rismhi3d` : perform RISM/HI calculation
- `ts4sdump` : decode the binary (or compressed) data of grid-based distributions that generated by `rismhi3d`
- `gmxtop2solute` : the tool to generate the solute forcefield file from the TOP file of GROMACS. The compilation of `gmxtop2solute` does not require GROMACS.
- `gensolvent` : the tool that helps to generate the solvent setting file

3.2 System requirement

System requirement:

- One of the following operating systems:
  - Linux 4 or above (4.15 recommended)
  - Mac OS 10.6 or above (10.14 recommended)
  - Windows Cygwin
  - Windows Subsystem Linux
- C++03 (e.g. GCC 4 or above, Apple LLVM 10 or above) (C++11 recommended)

Library requirement:

- **FFTW3 with double precision** (required)
- GROMACS (optional, for reading XTC)
- LIBZ (optional, for output compression)

3.3 Configuration

The configurations of this software is defined in `header.h`. A basic version can be built with an empty file of `header.h` (don't delete it). By adding one or more (or free combination) of the following options to `header.h`, you can build the software with different advanced features (i.e. parallel calculation, multithread FFTW, output compression, GROMACS XTC support):

- **#define _LOCALPARALLEL_** : enable paralleling. By default the paralleling will be done via multi processes.
- **#define _LOCALPARALLEL_PTHREAD_** : enable pthread paralleling. Require **_LOCALPARALLEL_** and pthread.
- **#define _LOCALPARALLEL_FFTW_** : enable manual paralleling of FFTW. Require **_LOCALPARALLEL_** (recommended if _LOCALPARALLEL_ is defined).
- **#define _FFTWMPPARALLEL_** : enable multithreading of FFTW. Require a multithread version of FFTW (not recommended).
- **#define _LIBZ_** : enable the compression of TS4S output (see Sec. 5 5). Require zlib;
- **#define _GROMACS_** : enable the XTC input (see Sec. 5 3)

As well as some other features that does not require additional packages:

- **#define _INTERACTIVE_** : enable the interactive mode (see Sec. 6 1). Require _LOCALPARALLEL_PTHREAD_ and pthread.
- **#undefine _TTYPROMPTCOLOR_** : undefine this flag to disable the color of error messages and underlines of directories and filenames in the screen outputs.

3.4 Compilation guidance

3DRISMHI requires the double precision version of FFTW3, so install FFTW3 before install 3DRISMHI. Define your FFTW3 path in $FFTWPATH:

```
export FFTWPATH=/opt/fftw-3.3.8 # if your FFTW is installed in /opt/fftw-3.3.8
```

Several kinds of 3DRISMHI can be built. Please choose one of the following sub sections to compile the source code:

3.4.1 The basic version

Single thread, no data compression, no XTC support.

`header.h` is empty

Then compile the four tools with:

```
g++ -O2 rismhi3d.cpp -o rismhi3d -I$FFTWPATH/include -L$FFTWPATH/lib -lfftw3 -lm
```

```
for toollname in ts4sdump gmxtop2solute gensolvent; do g++ -O2 $toolname.cpp -o $toolname -I$FFTWPATH/include
-L$FFTWPATH/lib -lfftw3 -lm; done
```

3.4.2 The fork parallel version

Parallel with forking of processes (use fork()), no data compression, no XTC support.

`header.h` contains the following content:

```
#define _LOCALPARALLEL_
#define _LOCALPARALLEL_FFTW_
```

Then compile the four tools with the same commands as the basic version:

```
g++ -O2 rismhi3d.cpp -o rismhi3d -I$FFTWPATH/include -L$FFTWPATH/lib -lfftw3 -lm
for toollname in ts4sdump gmxtop2solute gensolvent; do g++ -O2 $toolname.cpp -o $toolname -I$FFTWPATH/include
-L$FFTWPATH/lib -lfftw3 -lm; done
```

3.4.3 The pthread parallel version

Parallel with pthread or fork, no data compression, no XTC support. Require **pthread** library.

`header.h` contains the following content:

```
#define _LOCALPARALLEL_
#define _LOCALPARALLEL_FFTW_
#define _LOCALPARALLEL_PTHREAD_
```

Then compile the four tools with the same commands as the basic version:

```
g++ -O2 rismhi3d.cpp -o rismhi3d -I$FFTWPATH/include -L$FFTWPATH/lib -lfftw3 -lm -lpthread
for toollname in ts4sdump gmxtop2solute gensolvent; do g++ -O2 $toolname.cpp -o $toolname -I$FFTWPATH/include
-L$FFTWPATH/lib -lfftw3 -lm -lpthread; done
```

3.4.4 Multithread FFTW version

Parallel with both fftw_threads and pthread/fork. No data compression, no XTC support. Require **pthread** library and **a multithread version of FFTW**. (This version is not recommended)

`header.h` contains the following content:

```
#define _LOCALPARALLEL_
#define _LOCALPARALLEL_FFTW_
#define _LOCALPARALLEL_PTHREAD_
#define _FFTWMPPARALLEL_
```

Then compile the four tools with the same commands as the basic version:

```
g++ -O2 rismhi3d.cpp -o rismhi3d -I$FFTWPATH/include -L$FFTWPATH/lib -lfftw3 -lfftw3_threads -lm -lpthread
for toollname in ts4sdump gmxtop2solute gensolvent; do g++ -O2 $toolname.cpp -o $toolname -I$FFTWPATH/include
-L$FFTWPATH/lib -lfftw3 -lfftw3_threads -lm -lpthread; done
```

3.4.5 LIBZ enabled version

Enable data compression of TS4S files. A basic ZLIB enabled version has the following content of `header.h`:

```
#define _LIBZ_
```

Then compile the four tools with the same commands as the basic version:

```
g++ -O2 rismhi3d.cpp -o rismhi3d -I$FFTWPATH/include -L$FFTWPATH/lib -lfftw3 -lm -lz
```

```
for toolname in ts4sdump gmxtop2solute gensolvent; do g++ -O2 $toolname.cpp -o $toolname -I$FFTWPATH/include
-L$FFTWPATH/lib -lfftw3 -lm -lz; done
```

3.4.6 GROMACS 4 enabled version

Enable XTC support. Require **GROMACS 4.x**. Before the following compiling, first please source the shell script of GROMACS, or define $GMXLDLIB with the lib folder of gromacs:

```
export GMXLDLIB=/opt/gromacs4/lib # suppose your GROMACS is installed in /opt/gromacs4
```

A basic GROMACS 4 enabled version has the following content of `header.h`:

```
#define _GROMACS_
```

```
#define _GROMACS4_
```

And the compilation commands with GROMACS 4.x are:

```
g++ -O2 rismhi3d.cpp -o rismhi3d -I$FFTWPATH/include -L$FFTWPATH/lib -lfftw3 -lm -I$GMXLDLIB/../include -L$GMXLDLIB
-lgmx -ldl
```

```
for toolname in ts4sdump gmxtop2solute gensolvent; do g++ -O2 $toolname.cpp -o $toolname -I$FFTWPATH/include
-L$FFTWPATH/lib -lfftw3 -lm -lgmx -ldl; done
```

3.4.7 GROMACS 5+ enabled version

Enable XTC support. Require **GROMACS** $\geq$ **5**. Before the following compiling, first please source the shell script of GROMACS, or define $GMXLDLIB with the lib folder of gromacs:

```
export GMXLDLIB=/opt/gromacs5/lib # suppose your GROMACS is installed in /opt/gromacs5
```

A basic GROMACS 5+ enabled version has the following content of `header.h`:

```
#define _GROMACS_
```

And the compilation commands with GROMACS 5.x (or above) are:

```
g++ -O2 rismhi3d.cpp -o rismhi3d -I$FFTWPATH/include -L$FFTWPATH/lib -lfftw3 -lm -I$GMXLDLIB/../include -L$GMXLDLIB
-lgromacs -ldl
```

```
for toolname in ts4sdump gmxtop2solute gensolvent; do g++ -O2 $toolname.cpp -o $toolname -I$FFTWPATH/include
-L$FFTWPATH/lib -lfftw3 -lm -lgromacs -ldl; done
```

3.4.8 A complete version with all features enabled

Enable paralleling with pthread and fork, enable interactive mode, enable data compression with ZLIB, and enable XTC support with GROMACS 5 (or above). Requires **pthread**, **libz**, and **GROMACS 5+**. FFTW is required but a multithread version of FFTW is not required. Before the following compiling, first please source the shell script of GROMACS, or define $GMXLDLIB with the lib folder of gromacs:

```
export GMXLDLIB=/opt/gromacs5/lib # suppose your GROMACS is installed in /opt/gromacs5
```

The content of `header.h` is as follows:

```
#define _GROMACS_
#define _LOCALPARALLEL_
#define _LOCALPARALLEL_PTHREAD_
#define _LOCALPARALLEL_FFTW_
#define _LIBZ_
#define _INTERACTIVE_
```

Then compile the four tools with:

```
g++ -O2 rismhi3d.cpp -o rismhi3d -I$FFTWPATH/include -I$GMXLDLIB/../include -L$FFTWPATH/lib -L$GMXLDLIB -lfftw3 -lm -lz -lgromacs -ldl
for toolname in ts4sdump gmxtop2solute gensolvent; do g++ -O2 $toolname.cpp -o $toolname -I$FFTWPATH/include -I$GMXLDLIB/../include -L$FFTWPATH/lib -L$GMXLDLIB -lfftw3 -lm -lz -lgromacs -ldl; done
```

4 A step-by-step guidance of running 3DRISMHI

Step 1. prepare the solvent setting file

To run the 3DRISMHI to calculate the solvent distributions around given solutes, three input files are required: the solvent setting file, the solute forcefield file, and the solute conformations. Some solvent setting file can be found in the software library; and if our library doesn't have your solvent, please generate a new solvent setting file according to Sec. 5 1.

Now suppose you have `tip3p.gaff` in `$IETLIB` folder.

Step 2. prepare the solute conformation file

Solute conformation(s). The solute conformations can be either PDB, GRO or XTC (see Sec. 5 3 for details). The conformation(s) should not contain any solvent of the kind that you want to perform 3DRISMHI for. The solute is not required to be electrically neutralized.

Now suppose you have a small molecule `methylcyclopentane.pdb` or a protein `protein.pdb`, which may not be compatible with 3DRISMHI. There are two ways to change it to the 3DRISMHI compatible format. In the first way, you can do it youself by keeping only the lines that begin with "ATOM", and add one line to define the box vector (suppose the box is $30 \times 30 \times 30 \text{Å}^3$, and please ignore the rulers in gray color):

```
|--------+---------+---------+---------+---------+---------+---------|
CRYST1   30.000   30.000   30.000  90.00  90.00  90.00 P 1           1
|--------+---------+---------+---------+---------+---------+---------|
```

If you have GROMACS, this can be easily done with the following commands:

```
gmx editconf -f methylcyclopentane.pdb -o solute.gro -box 3 3 3 -center 1.5 1.5 1.5
gmx editconf -f solute.gro -o solute.pdb
```

Step 3. prepare the solute forcefield file

The solute forcefield file. The solute forcefield file can be translated from the GROMACS top file (see Sec. 5 2) with `gmxtop2solute`. Suppose you already have `solute.top`, the solute forcefield file `soltue.ff` can be generated with the following command:

```
gmxtop2solute -top solute.top -o solute.ff
```

If your `solute.top` depends on the GROMACS package, then please use $GMXDATA or -ffpath to include the forcefield predefinitions of GROMACS:

```
GMXDATA=/opt/gromacs/share gmxtop2solute -top solute.top -o solute.ff
```

or

```
gmxtop2solute -top solute.top -ffpath /opt/gromacs/share/gromacs/top -o solute.ff
```

(Please note that the number of atoms in the solute conformation file should be exactly the same as in the solute forcefield file.)

The GROMACS TOP file can be generated by either AMBER TOOLS + ACPYPE or GROMACS. If you have GROMACS:

```
gmx pdb2gmx -f protein.pdb -o solute.pdb -p solute.top
```

or if you have both AMBER Tools and ACPype:

```
antechamber -i solute.pdb -fi pdb -o solute.mol2 -fo mol2 -s 2 -c bcc
acpype.py -i solute.mol2 -c user
cp solute.acpype/solute_GMX.top solute.top
cp solute.acpype/solute_GMX.pdb solute.pdb
```

Step 4. Run the 3DRISMHI

A simple run of 3DRISMHI is shown as below, the 3DRISM-KH calculation of the TIP3P water around a solute molecule defined in output.ff and output.pdb:

```
rismhi3d -p $IETLIB/tip3p.gaff -s solute.ff -f solute.pdb -nr 60x60x60 -rc 1.2 -do-rismhi-kh -o output -rdf-grps 6-1,6-2,1-1,2-1 -rdf-bins 60 -log log.txt
```

For advanced users: the above command is equivalent to the following command:

```
rismhi3d -p $IETLIB/tip3p.gaff -s solute.ff -f solute.pdb -nr 60x60x60 -rc 1.2 -cmd hshi closure=kh rism report:energy display:rdf savee:cmd,guv -o output -rdf-grps 6-1,6-2,1-1,2-1 -rdf-bins 60 -log log.txt
```

(the green text are optional parameters and commands)

The screen output (or see `log.txt` if `-log log.txt` is specified) can be seen in Fig. 1 if no error occurs. At the same time, a file `output.ts4s` containing the command, density distributions and the HI density distributions is generated in the current folder. This file can be decoded with `ts4sdump`:

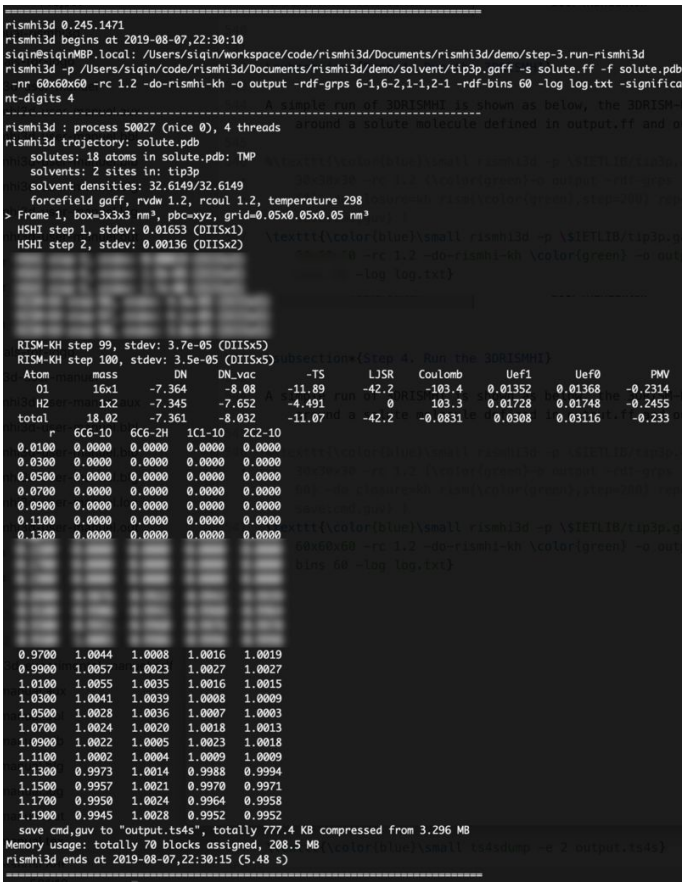
```
ts4sdump output.ts4s
ts4sdump -e 2 output.ts4s
ts4sdump -d 2 output.ts4s
```

For hydrophilic solutes, the only RISM need to be performed. For the above example, the 3DRISM-KH calculation can be performed with the following command:

```
rismhi3d -p $IETLIB/tip3p.gaff -s solute.ff -f solute.pdb -nr 60x60x60 -rc 1.2 -do-rism-kh -o output -rdf-grps 6-1,6-2,1-1,2-1 -rdf-bins 60 -log log.txt
```

Again, for advanced users: this command is equivalent to the following command:

```
rismhi3d -p $IETLIB/tip3p.gaff -s solute.ff -f solute.pdb -nr 60x60x60 -rc 1.2 -cmd closure=kh rism report:energy display:rdf savee:cmd,guv -o output -rdf-grps 6-1,6-2,1-1,2-1 -rdf-bins 60 -log log.txt
```



```
rismhi3d 0.245.1471
rismhi3d begins at 2019-08-07,22:30:10
siqin@siqinMBP.local: /Users/siqin/workspace/code/rismhi3d/Documents/rismhi3d/demo/step-3.run-rismhi3d
rismhi3d -p /Users/siqin/code/rismhi3d/Documents/rismhi3d/demo/solvent/tip3p.gaff -s solute.ff -f solute.pdb
 -nr 60x60x60 -rc 1.2 -do-rismhi-kh -o output -rdf-grps 6-1,6-2,1-1,2-1 -rdf-bins 60 -log log.txt -significa
nt-digits 4
------------------------------------------------------------------------
rismhi3d : process 50027 (nice 0), 4 threads
rismhi3d trajectory: solute.pdb
   solutes: 18 atoms in solute.pdb: TMP
   solvents: 2 sites in: tip3p
   solvent densities: 32.6149/32.6149
   forcefield gaff, rvdw 1.2, rcoul 1.2, temperature 298
> Frame 1, box=3x3x3 nm³, pbc=xyz, grid=0.05x0.05x0.05 nm³
  HSHI step 1, stdev: 0.01653 (DIISx1)
  HSHI step 2, stdev: 0.00136 (DIISx2)
  RISM-KH step 99, stdev: 3.7e-05 (DIISx5)
  RISM-KH step 100, stdev: 3.5e-05 (DIISx5)
  Atom      mass        DN      DN_vac        -TS       LJSR    Coulomb       Uef1       Uef0        PMV
    O1      16x1    -7.364      -8.08     -11.89     -42.2      -103.4    0.01352    0.01368    -0.2314
    H2       1x2    -7.345      -7.652    -4.491         0       103.3    0.01728    0.01748    -0.2455
 total     18.02    -7.361      -8.032    -11.07     -42.2     -0.0831     0.0308    0.03116     -0.233
     r   6C6-1O   6C6-2H   1C1-1O   2C2-1O
0.0100   0.0000   0.0000   0.0000   0.0000
0.0300   0.0000   0.0000   0.0000   0.0000
0.0500   0.0000   0.0000   0.0000   0.0000
0.0700   0.0000   0.0000   0.0000   0.0000
0.0900   0.0000   0.0000   0.0000   0.0000
0.1100   0.0000   0.0000   0.0000   0.0000
0.1300   0.0000   0.0000   0.0000   0.0000
0.9700   1.0044   1.0008   1.0016   1.0019
0.9900   1.0057   1.0023   1.0027   1.0027
1.0100   1.0055   1.0035   1.0016   1.0015
1.0300   1.0041   1.0039   1.0008   1.0009
1.0500   1.0028   1.0036   1.0007   1.0003
1.0700   1.0024   1.0020   1.0018   1.0013
1.0900   1.0022   1.0005   1.0023   1.0018
1.1100   1.0002   1.0004   1.0009   1.0009
1.1300   0.9973   1.0014   0.9988   0.9994
1.1500   0.9957   1.0021   0.9970   0.9971
1.1700   0.9950   1.0024   0.9964   0.9958
1.1900   0.9945   1.0028   0.9952   0.9952
  save cmd,guv to "output.ts4s", totally 777.4 KB compressed from 3.296 MB
Memory usage: totally 70 blocks assigned, 208.5 MB
rismhi3d ends at 2019-08-07,22:30:15 (5.48 s)
```

FIG. 1. The screen output (or in log.txt) of running 3DRISMHI

5 Input and output

3DRISMHI work with three input files and some options. The three input files are:

- Solvent settings (-p, -solvent, see Sec. 5 1): the information of te solvent
- Solute forcefield (-s, -solute, see Sec. 5 2): forcefield parameters of the solute
- Solute conformation(s) (-f, -traj, see Sec. 5 3): the atomic structure of solute molecules
- Parameters, options and commands (see Sec. 6): commands to run, number of grids, number of threads, etc.

The output of 3DRISMHI consists of screen outputs, TS4S outputs and RDF outputs:

- Screen outputs (see Sec. 5 4): brief reports and debug information
- TS4S files (see Sec. 5 5): grid based distributions
- RDF file (see Sec. 5 6): text file of RDFs, generated with **-cmd save:rdf**. (Note: RDF can be printed on screen with **-cmd report:rdf**)

Only a few screen output will be generated by default. More details of screen output, TS4S or RDF output will be generated when necessary (see Sec. 6)

5.1 Input: Solvent setting file

The solvent setting file is specified with -p or -solvent. If the current folder doesn't have the solvent setting file, the program will search for the path defined in $IETLIB. Normally the filename should not begin with "-", and if you do have such a file, use "--" before the filename (e.g. `-p -- -solvent_settings.gaff`)

The solvent setting file contains several sections: [solvent], [atom], [bond], and [gvv_map] ([gvv_map] is optional).

The **[solvent]** section mainly contains the following parameters:

- **ff**: the forcefield type specifier, can be: gaff/amber/opls. The mixing rule of VdW radiuses are arithmetic (-arith-sigma) in gaff/amber or geometric (-geo-sigma) in opls. The energy unit is kJ/mol (-Tdef 120.27) in gaff/opls or kcal/mol (-Tdef 502.97) in amber.

- **rvdw, rcoul, rc**: the cutoff distance (in nm) for the interactions. -rc will specify the cutoff for both -rcoul and -rvdw. The LJ interaction is calculated at a hard cutoff at -rvdw, while the Coulomb interaction can be calculated with either a PME swithed at -rcoul (if PME is enabled with -pme) or a hard cutoff at -rcoul (if PME is disabled by -nopme)

- **density** and **bulk-density**: a list of the densities or bulk densities of all solvent components. The unit is nm^{-3}. The bulk densities are the pure liquid densities of each solvent component.

- **gvv**: the solvent-solvent correlations. **gvv** is followed by the name of gvv file and the grid size, e.g. "-gvv 0.001 tip3p.gvv" or "-gvv tip3p.gvv 0.001" means that the gvv is defined in tip3p.gvv with the grid size of 0.001 nm. The gvv file contains a number of columns, each column is the correlation function of certain pair of solvent sites. The mapping between columns and solvent site pairs are defined by [gvv_map] section or in a default order (if [gvv_map] section is missing). The real mapping of gvv can be seen when running the program with "-list" option.

- **zeta**: the solvent-solvent zeta correlations, only used in HI calculations. The definition of zeta is similar to gvv, while the only difference is the unit of zeta is energy. zeta has no mapping between columns and molecule pairs, and should contain N$\times$N columns for N solvent molecules.

- **dielect** or **dipole**: a list of dielectric constants (vacuum is 1) or dipole moments (unit is $\mathrm{e}\cdot\mathrm{nm}$) of all molecules. The dielectric constants and dipole moments are optional and only used in certain algorithms (e.g. -Coulomb dielect, see Sec. 6 for details)

The **[atom]** section defines all the atoms. The [atom] section consists seven columns (more columns are ignored), separated by spaces or tabs:

column	definition
1	atom name
2	moleucle name
3	index
4	group index
5	charge
6	sigma (nm)
7	epsilon

The group index is the site index, and the atoms with the same group (or site) index can be treated as one site in RISM calculation. It is highly suggested that completely equivalent atoms are grouped into one site, which will greatly reduce the computational cost.

The **[bond]** section defines the bonds and pairs of each pair of atoms. The **[bond]** section consists of three or four columes, where the first two columns are the two atoms, and the third column is the bond length or fixed pair distance between the pair of atoms (unit: nm). The fourth column is the RMSD of the fluctuations of bond lengths (or pair distances), which is optional and will be treated as 0 if the fourth column is missing. Warning: don't define one bond/pair twice.

5.2 Input: Solute forcefield

The solute forcefield file is specified with -s or -solute. This file contains only **[solute]** section(s). The **[solute]** section contains six or eight (more column are ignored) columns:

column	6-col format	8-col format
1	atom name	atom index
2	moleucle name	atom name
3	atom mass	residue id
4	partial charge	moleucle name
5	sigma (nm)	atom mass
6	epsilon	partial charge
7		sigma (nm)
8		epsilon

The solute forcefield can be simply translated from the GROMACS top file with the “gmxtop2solute” tool provided in the software package.

5.3 Input: Solute conformation(s)

The solute conformation(s) are defined in the trajectory file, specified with -f or -traj. The trajectory file can be a PDB, GRO or XTC file.

The PDB file: only lines begin with “ATOM” or “CRYST1” will be processed. The number of atoms defined by the ATOM lines should be consistent with the solute forcefield file. The box size should be defined with a CRYST1 line. The PDB file can contain multiple frames, separated by “ENDMDL”.

The GRO file can also contain multiple frames.

The XTC file can be processed only when the software is compiled with _GROMACS_ options. It’s fine to turn off all _GROMACS_ flags when compling the software, as long as you don’t have GROMACS or don’t want to be bothered by this feature. The frames of XTC between the time (in ps) defined in **-b** and **-e** are handled, and **-dt** specifies the time interval between frames that handled in 3DRISMHI.

5.4 Output: screen report

The screen output consists of the running information and some brief reports of calculation results.

The running information can be muted with **-v 0**. If you want to trace the running details, please use **-v 1**, **-v 2** or simply **-v**.

More running information is shown in debug mode. **-debug 0** will mute all debug message, and **-debug 1** will allow to show some important messages (e.g. real location of input files, allocated memory, thread, etc.). In **-debug 2**, a detailed running process is printed on screen, including calling of major functions in the source code. Further in **-debug 3** or **-debug-crc**, the CRC check sum of important memory bulks will be displayed on the screen. Please note that -debug 3 or -debug-crc will perform the CRC calculation, which will require additional computing time.

A detail report of time consumption will be displayed at the end with **-v 2** or **-debug 1**/**2**/**3**.

The brief reports of calculation results will be displayed according to the command defined in the command queue. Please see Sec. 6 for details of commands and the command queue. For example, if you want a detailed report of energy or correlation functions, you need to add a “-cmd report:energy” or “-cmd report:cuv” command; if you want to show the rdf, you need to add a “-cmd display:rdf” command.

The screen output can be redirected to a log file specified with **-log** (**-log screen**, **-log stdout** and **-log stderr** will redirect the screen output to stdout or stderr).

5.5 Output: TS4S file

The TS4S file contains one or more frames of 4D tensors. Normally in 3DRISMHI, the 4D tensors can be LJ potentials, Coulomb potentials, total correlation, direct correlation, total density profile and HI density. To generate TS4S output, please specify **-cmd save:...** in the commands.

The name of the output TS4S file is specified with **-o[v][0/1/2]** or **-a[0/1/2]**, where "ov" represepts overriding, and "a" represents appending. The TS4S data can be uncompressed (**-o[v]0** or **-a0**) or compressed with ZLIB (if you compiled with the _ZLIB_ option). Please note that the TS4S file may be extrodinarily huge, and the **-significant-digits** or **-sd** option may be helpful for higher compressibility with lower accuracy of output data (e.g. **-sd 5** will keep only five significant digits, and **-sd float** will trim all output data to float).

The TS4S output will be performed with the "save" command defined in the command queue (see Sec. 6 for details). The TS4S file will be generated upon the first time of saving. The filename extension of TS4S is always ".ts4s". If the filename is not specified, a default filename solute.solvent.YYMMDD_HHMM.ts4s will be used.

Please use "ts4sdump" (provided in the software package) to check and decode the TS4S file:

```
ts4sdump filename.ts4s # list all frames of filename.ts4s

ts4sdump -check filename.ts4s # check the CRC checksum of each frame in filename.ts4s

ts4sdump -d 2 filename.ts4s # decode the second frame of filename.ts4s

ts4sdump -e 3 filename.ts4s # decode the third frame of filename.ts4s, print with grid coordinates
```

(For developer:) The file format of a TS4S file is defined in compress.cpp of the source code. TS4S consists of a number of data blocks, each block corresponding to a 4D tensor. One 4D tensor block begins with the IETSPage-Header structure, followed by comment text and tensor data. The decoded 4D tensors are organized in the order of "tensor[solvent_site][z][y][x]".

5.6 Output: RDF file

The RDF can be calculated and displayed or written to a text file. `rismhi3d.cpp` can perform the RDF calculations if **-rdf-grps** are defined and **-cmd display:rdf** or **-cmd display@end:rdf** is specified.

The RDF groups are defined in "**-rdf-grps** u1-v1,u1-v2,...", where u1, u2 ... stand for the indexes of solute atoms and v1, v2 ... stand for the indexes of solvent sites. The number of RDF bins is defined in "**-rdf-bins** 50", where 50 is the default bin number. The RDF will be performed up to the distance defined in "**-rc**". For an advanced user who wants to see the RDFs of HI densities or direct correlations, the "**-rdf-content**" option can be used to specify the RDF to calculate: rdf (density profile, by default), h (direct correlation), dd (HI density), c (direct correlation), ch (ch=$c * h$), lj (LJ potential) or coul (Coulomb potential).

The displaying or saving of RDF is performed in "-cmd display:rdf" or "-cmd save:rdf". The RDF is calculated when necessary, so no RDF calculation will be performed if the RDF is not displayed or saved. "-cmd display:rdf" will display the RDF on screen (or -log file), while "-cmd save:rdf" will save the RDF to a text file. The name of the RDF file begins with the name of TS4S file, and ended with a ".rdf" extension.

6 Parameters, options and commands

6.1 Parameters for running

In the command line parameters, you can specify the running parameters, tell the software what to compute and display, and override some solvent settings. (Please note that all the settings in the **[solvent]** section can be overridden by the command line parameters.) Here is a list of the major parameters for running:

- **-nt** and **-np**: number of parallel runnings. -np 1 or -nt 1 will disable the paralleling. **-nt** will use pthread while **-np** will use fork. Please note that paralleling is still possible (with -np) even if you hate or don't have pthread. **-np** is enabled with _LOCALPARALLEL_ option in compiling, while **-nt** requires both _LOCALPARALLEL_ and _LOCALPARALLEL_PTHREAD_ in compiling. Note: the 3DRISMHI will automatically set -nt to the maximum number of your CPU cores. So don't forget to change -nt or -np if you are running 3DRISMHI on a cluster.

  An important thing of paralleling is that the _LOCALPARALLEL_FFTW_ will manually parallelize the FFTW by performing multiple FFTWs at the same time. For an N site (or molecule) system, N$\times$N FFTWs need to be performed in one RISM (or HI) iteration, thus the maximum number of working threads in _LOCALPARALLEL_FFTW_ mode is N$\times$N (more threads will be idling). For example, if your solvent contains nothing but water (one molecule and two sites), then only one thread/process will be busy in the HI convolutions and at most four threads/processes will be working in the RISM convolutions.

  Other parts of massive computations (i.e. building forcefield and performing DIIS) will make full use of multi threads/processes.

- **-ntf**: number of threads for FFTW. This is enabled with multithread FFTW. This opention does little or negligible improvement of the computation efficiency.

- **-nice**: the nice level of running this software.

- **-nr** : the grid number. It can be one number for both three dimensions; three numbers for X, Y and Z respectively; or NXxNYxNZ. e.g. "-nr 50" is equivalent to "-nr 50 50 50" and "-nr 50x50x50". The grids of X, Y and Z can be different, e.g. "-nr 50x60x70". The grid numbers are suggested to be even numbers.

- **-interact**: enable the interactive mode. Pressing enter at running will halt the calculation, and you can print the report, change parameters, continue running, end current RISM/HI calculation or quite the program. Warning: don't enable the interactive mode when you are running 3DRISMHI in background (e.g. nohup or on a cluster).

- **-do-rism-kh**, **-do-rismhi-kh** and **-do-rismhi-d2**: perform 3DRISM or 3DRISMHI with the KH or D2MSA closures. The RDFs will be displayed on screen if **-rdf-grps** are defined, while a TS4S file containing the running command and the density distributions will be generated if the output file is specified.

- **-cmd** or **-do**: the command(s) you want to run. The software will do nothing if the command queue is empty. See Sec. 6 2 for details.

- **-skip-missing-xvv** or **-skip-zero-xvv**: skip the convolution related to fully zero solvent-solvent correlations (including wvv, nhkvv and zeta). This option is disabled by default, and can be enabled by **-skip-missing-xvv** [yes] or **-skip-zero-xvv** [yes]. This option should yield exactly the same results (including the CRC checksum) as not enabled while significantly faster as fewer FFT is required to be performed.

And here is a list of some important **[solvent]** settings that can be redefined:

- **-temperature** 298: specify the temperature in Kelvin

- **-Tdef** 120.27: specify the temperature of defining the energy unit. 1 kJ/mol = 120.27 K, 1 kcal/mol = 502.97 K. Can be specified with **-ff**.

- **-arith-sigma** and **-geo-sigma**: set the combing rule of VdW raiuds to arithmetic or geometric averaging.

- **-density** and **-bulk-density** can be redefined in the command line parameters.

- **-ndiis**, **-ndiisrism** and **-ndiishi**: the maximum steps of DIIS. More steps of DIIS will have better convergence of self-consistent-field iterations while require more memory.

- **-delvv**, **-delrism** and **-delhi**: the step in factor of the self-consistent-field iterations. **-delvv** for both RISM and HI. Both are 1 by default, and 0.7 is recommended by RISM.

- **-errtol**, **-errtolrism** and **-errtolhi**: the error tolerance of convergence. Both are 10^{-12} by default. Although this is fine for HI, the RISM iterations have little chances to reach the error of 10^{-12}. This number is suggested to be 10^{-7} in AMBER RISM.

- **-bound-to-ram** or **-ignore-memory-capacity**: by default, the software will detect the capacity of the physical memory, and will terminate when the memory is exceeding the physical memory. This feature can be disabled by **-ignore-memory-capacity**, which will cause extremely low computational efficiency as well as high risk to damage your hard disk. Ignore the memory capacity check only when you know what you are doing.

Other advanced options can be seen in Sec. 6 3

6.2 Commands and command queue

The command forms a command queue, and will be performed one by one after the frames are read from the trajectory file. The command(s) specified in **-cmd** will be added to the command queue. You can use to relocate the current command instead of inserting it to the end of the queue, e.g. "-cmd report@5:energy" will insert "-cmd report:energy" to the 5th command (the command queue begins with 1). b and e can be used to force the command to runs before handling of any frame or after handling all frames, e.g. "-cmd report:rdf" will report the RDF at each frame, and "-cmd report@end:rdf" will report the overall RDF after all frames have been processed.

Followings are some basic commands for setting closures, running HI/RISM calculations, and generating reports or output files of the output results:

- **build-ff**: **-cmd build-ff**: force to rebuild the forcefield. The forcefield is built automatically with **rism** or **hshi**, and this command is used only when you need to do something without perfoming RISM or HI.
- **rism**: **-cmd rism**,step=100: perform 3DRISM with specified number of maximum steps and the closure(s) defined in the **closure** command before.
- **ssoz**: **-cmd ssoz**,step=100: perform unrenormalized 3DRISM with specified number of maximum steps and the closure(s) defined in the **closure** command before. The unrenormalized 3DRISM is the very original version of 3DRISM, which has big issues in electrostatic interactions.
- **hshi**: **-cmd hshi**,step=100: perform HI with specified number of maximum steps.
- **closure**: **-cmd closure=**closure_A[,closure_B,...]: set the closure(s) for each molecule. Different sites can be calculated with different closures, and at most 20 closures are allowed to be specified here. Particularly, all the sovlent sites will use the same specified closure if only one closure is given here. This version of 3DRISMHI support all the closures mentioned in Sec. 2 3: HNC, MSA, KGK, PLHNC, KH, PSE2, PSE3, ... PSE10, PY, HNCB, D2, MS, MSHNC, BPGGHNC, VM, MP.
- **closure-a**: set the closure(s) for each site instead of molecule.
- **closure-factor**: **-cmd cf=**cf_A[,cf_B,...]: set the extra parameters that used in closures.
- **report**: **-cmd report:**Euv/energy/cuv/rdf: generate a report on screen (or to -log file). **-cmd report:Euv** will display a brief report of total energies, while **-cmd report:energy** will display a detailed report of total energies. **-cmd report:cuv** will display the total direct correlations, and **-cmd report:rdf** will display the RDFs if you have defined the RDF groups. In addition, **-cmd report:energy,cuv** is equivalent to **-cmd report:all**.
- **display**: **-cmd display:**lj/coul/Euv/energy/cuv/dN/dN0/TS/GGF/rdf: display the values of the chosen variables
- **save**: **-cmd save:**cmd/lj/coul/cuv/huv/hlr/dd/ddp/nphi/guv/rmin/rdf: save the specified quantity. **-cmd save:cmd** will save the command line arguments to the TS4S file, and **-cmd save:guv** will save the density distributions of each solvent site at each spacial grid to the TS4S file. **-cmd save:lj,coul,cuv,huv,hlr,dd** will save LJ potentials, Coulomb potentials, direct correlations, total correlations, long range total correltations, HI density to the TS4S file, and **-cmd save:rmin** will save the minimal-to-solute distances to the TS4S file. Additionally, **-cmd save:rdf** will save the RDF to the RDF file (which is a text file).
- **savee**: mostly the save as **save**. The only difference is that **savee** will perform saving only when the output TS4S file is explicitly specified, while **save** will use an default filename if the output TS4S file is not explicitly specified.

Addtionally, some shortcuts can be used to add a bundle of commands. Don't use two or more shortcuts, otherwise both the two sets of commands will be performed.

- **-do-rism-kh** = -cmd closure=kh rism report:energy display:rdf savee:cmd,guv
- **-do-rismhi-d2** = -cmd hshi closure=d2 rism report:energy display:rdf savee:cmd,guv
- **-do-rismhi-kh** = -cmd hshi closure=kh rism report:energy display:rdf savee:cmd,guv

6.3 Other options

Below is a list of major advanced options that can be defined in both the **[solvent]** section of the solvent setting file or the command line parameters:

- **-lsa** and **-lsb** for HI: the two parameters, A and B of the liquid equation of state[1–3]. B is automatically computed, and A can be defined with **-lse-a** or **-lsa**. The recommended values of A can be seen from previous experimental measurements[8].
- **-theta** for HI: define the energy cutoff (in kT), above which the regions are treated as hard cores or no solvent regions. The default cutoff is 5 kT.
- **-Coulomb**: the preprocessing algorithm of the Coulomb interactions in HI and RISM. Can be: none (=-Coulomb), dielect, or Yukawa.
- **-Yukawa** 0.5: the same as **-Coulomb Yukawa** 0.5. 0.5 here is the characteristic length (unit: nm) of the exponential function of the Yukawa potential. The Debye length of homogeneous bulk liquid will be used if the characteristic length is not specified. The dielectric constant for the Yukawa potential is defined in **-dielect-y**.
- **-rb** or **-Bohr-radius**: the minimal raiuds of an atom. By default it is 0.052911 (nm).
- **-sd** or **-significant-digits**: the significant digits in the TS4S files, can be "float", "double" or a number (significant digits in decimal)
- **-closure-enhance-level** 1 or **-enhance-closure** 1: scale down the changes of SCF iterations with $(1+h^2)^{\alpha/2}$, where α is the closure enhancement level. This option greatly improves the convergence of the self-consistent-field iterations and is turned on by default. Use **-no-closure-enhance** to turn this feature off.
- **-bounded-to-ram**: don't exceed the physical memory capacity when using memory. This feature is on by default, and use **-ignore-memory-capacity** to turn this feature off.
- **-xvv-extend** 0: extend the **gvv** of solvent. If the input gvv contains the RDFs of 2 nm, then -xvv-extend 5 will extend it to 10 nm by filling the extended regions with 1. This option is helpful when you **gvv** is poor, but don't expect too much.

[1]S. Cao, F. K. Sheong, and X. Huang, "Reference interaction site model with hydrophobicity induced density inhomogeneity: An analytical theory to compute solvation properties of large hydrophobic solutes in the mixture of polyatomic solvent molecules," Journal of Chemical Physics **143**, 054110 (2015).
[2]S. Cao, L. Zhu, and X. Huang, "3DRISM-HI-D2MSA: an improved analytic theory to compute solvent structure around hydrophobic solutes with proper treatment of solute–solvent electrostatic interactions," Molecular Physics **116**, 1003–1013 (2018).
[3]S. Cao, K. A. Konovalov, I. C. Unarta, and X. Huang, "Recent Developments in Integral Equation Theory for Solvation to Treat Density Inhomogeneity at Solute–Solvent Interface," Advanced Theory and Simulations **0**, 1900049 (2019).
[4]E. L. Ratkova, D. S. Palmer, and M. V. Fedorov, "Solvation Thermodynamics of Organic Molecules by the Molecular Integral Equation Theory: Approaching Chemical Accuracy," Chemical Reviews **115**, 6312–6356 (2015).
[5]T. Luchko, S. Gusarov, D. R. Roe, C. Simmerling, D. A. Case, J. Tuszynski, and A. Kovalenko, "Application of the srism approach to the study of the capacitance of the double layer of a high density primitive model electrolyte," Journal of Chemical Theory and Computation **6**, 607 (2010).
[6]S. Gusarov, B. S. Pujari, and A. Kovalenko, "Efficient treatment of solvation shells in 3d molecular theory of solvation," Journal of Computational Chemistry **33**, 1478–1494 (2012).
[7]P. Pulay, "Convergence acceleration of iterative sequences. the case of scf iteration," Chemical Physics Letters **73**, 393 (1980).
[8]J. DYMOND and R. MALHOTRA, "THE TAIT EQUATION - 100 YEARS ON," INTERNATIONAL JOURNAL OF THERMOPHYSICS **9**, 941–951 (1988).