ANATRA 1.0 User Guide

Kento Kasahara@Osaka University

Jun 15, 2025

CONTENTS:

CHAPTER

ONE

WHAT IS ANATRA?

ANATRA (*Ana*lyze *Tra*jectories) is a collection of Tcl/Fortran90 programs for analyzing trajectories obtained from Molecular Dynamics (MD) simulations.

We often need to analyze specific atoms, molecules, or amino acid residues of proteins in a system. However, writing a general-purpose program to extract such specific parts is not an easy task. This is because it is necessary to extract atom groups that satisfy multiple complex conditions simultaneously, such as:

> "residues numbered from 11 to 50, with residue name ALA, and only heavy atoms."

Creating a program that can interpret and extract such sets defined by intersections and unions of multiple conditions can be quite difficult.

On the other hand, **VMD** (*Visual Molecular Dynamics*), a tool widely used by MD users around the world, offers an excellent feature called **Atomselection** for selecting specific parts of a system. For instance, the above condition can be expressed in VMD′s Atomselection syntax as:

```
resid 11 to 50 and resname ALA and noh
```

which is easy to understand intuitively.

ANATRA adopts a design that leverages VMD′s Atomselection feature. As a result, users can use exactly the same syntax as in Atomselection for selecting specific parts of the system, eliminating the need to learn a new selection language—especially convenient for existing VMD users.

ANATRA consists of numerous Tcl/Fortran programs tailored to different types of analyses. To manage these programs in a unified way, a command-line interface named `"anatra"` is provided. Users can perform various analyses by executing:

```
$ anatra analysis_mode -option1 -option2 ...
```

In this workflow, the corresponding Tcl script is executed on VMD according to the specified analysis mode, utilizing Atomselection to identify the region of interest. The selected information is then passed as input to a Fortran program for analysis. All of this is handled internally, and users do not need to be aware of the intermediate steps—everything runs in a streamlined manner. At the same time, since the Fortran programs themselves do not implement Atomselection directly, the source code remains simpler and easier to maintain. This also allows users to create new analysis programs with ease by building upon the libraries provided by **ANATRA**.

- **Project Leader**
  - Kento Kasahara (Graduate School of Engineering Science, Univ of Osaka)
- **Contributor**
  - Ren Masayama (Graduate School of Engineering Science, Univ of Osaka)
  - Kazuya Okita (Graduate School of Engineering Science, Univ of Osaka)
  - Yuya Matsubara (Graduate School of Engineering Science, Univ of Osaka)
  - Yusei Maruyama (Graduate School of Engineering Science, Univ of Osaka)
  - Ryo Okabe (Graduate School of Engineering Science, Univ of Osaka)
  - Yuki Yamashita (Graduate School of Engineering Science, Univ of Osaka)
  - Nobuyuki Matubayasi (Graduate School of Engineering Science, Univ of Osaka)

- **xdrfile-1.1.4**

  https://ftp.gromacs.org/pub/contrib/

- **xdf.F90**
  This routine was originally developed by Wes Barnett and distributed under the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
  https://github.com/wesbarnett/libgmxfort
  Modified version of the routine was developed by Kai-Min Tu.
  https://github.com/kmtu/xdrfort

- **mt19937.f**
  This routine was developed by Makoto Matsumoto and Takuji Nishimura and was distributed under the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
  https://www.math.sci.hiroshima-u.ac.jp/m-mat/MT/VERSIONS/FORTRAN/fortran.html

CHAPTER

TWO

INSTALLING ANATRA

This chapter describes how to install ANATRA.

2.1 Required Software for Using ANATRA

- VMD 1.7.3 or later
- Intel Compiler (Intel OneAPI)
- bash

This manual assumes that bash is used as the login shell.[1]

VMD can be downloaded for free from:

```
https://www.ks.uiuc.edu/Development/Download/download.cgi?PackageName=VMD
```

Note that account registration is required.

Let us give a brief guide to installing VMD. After downloading `vmd-XXX.tar.gz` from the website, extract it to create a directory named `vmd-XXX`, and navigate into it. Open the `configure` file in a text editor, and set the `install_bin_dir=` at the top of the file to the desired installation directory. Then, on a Linux machine, run:

```
$ cd vmd-XXX
### Modifying configure manually ###
$ ./configure LINUXAMD64
```

Next, move into the `src` directory and run:

```
$ cd src
$ make install
```

[1] However, as described in the next section, ANATRA can also be used with other shells like `zsh` by manually modifying the corresponding rc file (e.g., `~/.zshrc`).

This will place the vmd binary in the directory specified by `install_bin_dir=`. Add this directory to your PATH to complete the installation.

2.2 Installing from Source

This section explains how to install ANATRA from source code.
The latest source code is maintained on GitHub:

```
https://github.com/kenkasa/anatra-release
```

First, clone the repository from GitHub.
Assuming you want to install ANATRA in `/home/USER/software`, navigate to that directory and run `git clone`:

```
$ cd /home/USER/software
$ git clone https://github.com/kenkasa/vmd_scripts.git
$ cd anatra
$ ls
README.md
install.sh
bin/
tcl/
f90/
```

If you have previously installed ANATRA, your `~/.bashrc` may already contain the following three lines. If so, delete them **before** proceeding, unless the `$ANATRA_PATH` variable already points to the same installation directory (if you' re unsure, it' s safe to remove them):

```
export ANATRA_PATH=...
export PATH=$ANATRA_PATH/bin:$PATH
export LD_LIBRARY_PATH=$ANATRA_PATH/f90/lib/external/netcdf/netcdf/lib:$LD_LIBRARY_PATH
```

Run `install.sh` to start the installation. You can specify the compiler to use as an argument to `install.sh`:

```
$ ./install.sh <gcc or intel>
```

By default, `intel` is selected. If your system supports the Intel compiler, it is recommended to use it, as some programs rely on Intel Math Kernel Library (MKL). When using `gcc`, the compilation of those programs will be skipped during installation.

During the build process, progress will be printed to the terminal.
If you see the following message near the end, it indicates that all Fortran programs have been successfully built:

```
Installation of ANATRA fortran programs have been
succesfully finished!!
```

If the `$ANATRA_PATH` variable is correctly set and the `anatra` command is available in your `PATH`, the installation was successful:

```
$ echo $ANATRA_PATH
/home/USER/software/anatra

$ which anatra
/home/USER/software/anatra/bin/anatra
```

If you are using a shell other than bash, add the equivalent lines to rc file for your shell. For example, for `zsh`:

```
export ANATRA_PATH=/path/to/vmd_scripts/anatra
export PATH=$ANATRA_PATH/bin:$PATH
export LD_LIBRARY_PATH=$ANATRA_PATH/f90/lib/external/netcdf/netcdf/lib
```

CHAPTER

THREE

USAGE OF ANATRA

3.1 Introduction

anatra is a post-processing analysis suite that combines the use of VMD and Fortran programs. This chapter provides a detailed guide on how to use **anatra**. As described previously, the standard workflow in **anatra**-based analysis proceeds in two stages. First, atom or residue selection and preprocessing are carried out using VMD, resulting in intermediate files. These files are subsequently analyzed via Fortran executables.

A list of available analysis modes can be obtained by executing the following command:

```
$ anatra -h
Available analysis in ANATRA
trjconv             (trj)   : Convert trajectory
distance            (dis)   : Distance Analysis
center_of_mass      (com)   : Center-of-Mass Analysis
radial_distr        (rdf)   : Radial-Distribution Analysis
spatial_distr       (sdf)   : Spatial-Distribution Analysis
rmsd                (rmsd)  : Root-Mean-Square-Deviation Analysis
interaction_energy  (ene)   : Interaction Energy Analysis
lipid_order         (scd)   : Lipid S_CD order-parameter Analysis
area_per_lipid      (apl)   : Area Per Lipid Analysis
z_profile           (zprof) : Z-profile Analysis
z_orient            (zori)  : Orientation Analysis along z
species_info        (spec)  : Get Species information
```

For example, the `trjconv` analysis mode may be invoked as follows:

```
$ anatra trjconv -option1 ...
```

Abbreviated mode names, as shown in parentheses (e.g., `trj` for `trjconv`), can also be used:

```
$ anatra trj -option1 ...
```

To display available options and example usage for each mode, one can execute:

```
$ anatra <analysis_mode> -h
```

This help message is generally sufficient for basic usage. However, for more advanced configurations or clarification, this manual should be consulted.
An example help output for `trjconv` is shown below (VMD-related output is omitted for brevity):

```
$ anatra trjconv -h
============================================================


                    Trajectory Convert


============================================================
Usage:
anatra trjconv                                                        \
  -stype      <structure file type>                                   \
  -sfile      <structure file name>                                   \
  -tintype    <input trajectory file type>                            \
  -tin        <input trajectory file name>                            \
  -totype     <output trajectory file type>                           \
  -to         <output trajectory file name>                           \
  -beg        <first frame to be read> (default: 1)                   \
  -end        <last frame to be read>                                 \
              (default: 0, corresponding to the final frame)          \
  -selX       <X-th VMD selection> (X = 0,1,2...)                     \
  -fit        <whether to apply fitting (true or false)>              \
  -centering  <whether to apply centering (true or false)>            \
              (default: false)                                        \
  -wrap       <whether to apply wrapping (true or false)>             \
              (default: false)                                        \
  -wrapcenter <wrapping center: origin or com (default: fragment)>    \
  -wrapcomp   <wrapping unit: residue, segid, chain, fragment, or nomp>   \
              (default: fragment)                                     \
  -refpdb     <reference PDB file name>                               \
              (required if fit = true)                                \
  -outselid   <selection ID for output molecules>                     \
  -fitselid   <selection ID for fitting or centering>                 \
  -refselid   <selection ID for reference>

Remark:
o If both fit = true and wrap = true are specified, wrapcomp is automatically set to 'com'
```

(continues on next page)

(continued from previous page)

```
Example (fitting):
anatra trjconv                              \
  -stype      parm7                         \
  -sfile      str.prmtop                    \
  -tintype    dcd                           \
  -tin        inp.dcd                       \
  -totype     dcd                           \
  -to         out.dcd                       \
  -beg        1                             \
  -end        150                           \
  -sel0       not water                     \
  -sel1       resid 1 to 275 and name CA \
  -sel2       resid 1 to 275 and name CA \
  -fit        true                          \
  -centering  false                         \
  -wrap       true                          \
  -wrapcenter origin                        \
  -wrapcomp   fragment                      \
  -refpdb     ref.pdb                       \
  -outselid   0                             \
  -fitselid   1                             \
  -refselid   2
```

As illustrated above, **anatra** accepts a considerable number of options, making it well-suited for use within `bash` scripts. An example script is provided below:

```
#!/bin/bash

anatra trjconv            \
  -stype    parm7         \
  -sfile    str.prmtop    \
  -tintype  dcd           \
  -tin      INP.dcd       \
  -totype   dcd           \
  -to       OUT.dcd       \
  -sel0     resid 1 to 20 \
  -outselid 0
```

3.2 Common Options

While **anatra** offers various analysis modes, many of them share a common set of command-line options. A summary of these shared options is presented below, followed by detailed descriptions of each.

Option	Function	Example
`-stype`	Specifies the structure file type	`-stype parm7`
`-sfile`	Specifies the structure file name	`-sfile A.prmtop`
`-tintype`	Specifies the input trajectory file type	`-tintype dcd`
`-tin`	Specifies the input trajectory file name	`-tin in.dcd`
`-flist_traj`	Specifies a file listing multiple trajectory files	`-flist_traj list.txt`
`-totype`	Specifies the output trajectory file type	`-totype dcd`
`-to`	Specifies the output trajectory file name	`-to out.dcd`
`-sel`x	Specifies a selection using VMD syntax	`-sel0 resid 1 to 20`
`-mode` or `-mode`x	Specifies how to group the selected atoms	`-mode residue`
`-fhead`	Header for output file names	`-fhead out`
`-prep_only`	If true, only preprocessing is performed	`-prep_only true`

- **-stype <structure file type>**
  - Default: N/A
  - Example: `-stype parm7`

  Specifies the structure file type. Structure files contain information such as atom types and possibly bonding details. Since many trajectory formats (e.g., `dcd` or `xtc`) store only atomic coordinates, a structure file is required for analysis. Most of the structure file types supported by VMD can be specified. Representative examples include:

  - **pdb**: PDB file
  - **gro**: GRO file
  - **psf**: CHARMM PSF file
  - **parm7**: AMBER parameter file (usually with extension `.prmtop`)

  Among these, `psf` (for CHARMM) and `parm7` (for AMBER) provide the most complete information and are recommended when available.

- **-sfile <structure file>**
  - Default: N/A
  - Example: `-sfile complex.prmtop`

Specifies the name of the structure file.

- **-tintype <input trajectory file type>**
  - Default: `dcd`
  - Example: `-tintype dcd`

  Specifies the type of the input trajectory file. Currently, with the exception of `trjconv`, supported formats are limited to `dcd`, `xtc`, and `netcdf`. `trjconv` supports all trajectory formats recognized by VMD.

- **-tin <input trajectory file>**
  - Default: N/A
  - Example: `-tin INP.dcd`

  Specifies the filename of the input trajectory. When using multiple files, list them as: `-tin run1.dcd run2.dcd`. Wildcards (e.g., `run*.dcd`) are also supported. Alternatively, you can use the `-flist_traj` option described below.

- **-flist_traj <text file>**
  - Default: N/A
  - Example: `-flist_traj trajlist.txt`

  Specifies a text file listing the input trajectories. If both `-tin` and `-flist_traj` are given, `-flist_traj` takes precedence. The file should contain one trajectory filename per line, as shown below:

```
traj1.dcd
traj2.dcd
traj3.dcd
```

- **-totype <output trajectory file type>**
  - Default: `dcd`
  - Example: `-totype dcd`

  Specifies the type of the output trajectory file. As with input, all VMD-supported formats can be used in `trjconv`. Other modes support only `dcd` and `xtc`.

- **-to <output trajectory file>**
  - Default: N/A
  - Example: `-to OUT.dcd`

Specifies the name of the output trajectory file.

- **-sel**$x$ **<selection>** ($x = 0, 1, \cdots$)
  - Default: N/A
  - Example: `-sel0 resid 1 to 20 and name CA`

  Specifies an atomic selection using VMD′s Atomselection syntax. The index $x$ is referred to as the selection ID (*selid*). Most analysis modes perform calculations on the region specified by each `-sel$x$`. In some cases, you must explicitly assign which `selid` is used for which purpose (e.g., `-outselid` in `trjconv`).

- **-mode**$x$ **<residue or whole or atom>** ($x = 0, 1, \cdots$)
  - Default: Mode-dependent
  - Example: `-mode0 residue`

  Specifies how to group the atoms selected by `-sel$x$` (see Fig. 3.1). Some analysis modes only require a single selection, using `-mode` to define grouping. Others require multiple selections, each needing a corresponding `-mode$x$`. See the mode-specific descriptions for details.

  - **residue**: Treat each residue in the selection as a separate molecule.
  - **whole**: Treat the entire selection as a single molecule.
  - **atom**: Treat each atom in the selection as an individual molecule.

  For example, if `-sel0 resname LIG` selects multiple ligand molecules and you wish to compute each center of mass, set `-mode residue`. Conversely, to treat a full protein selected via `-sel0 protein` as one object (e.g., for center of mass tracking), set `-mode whole`.

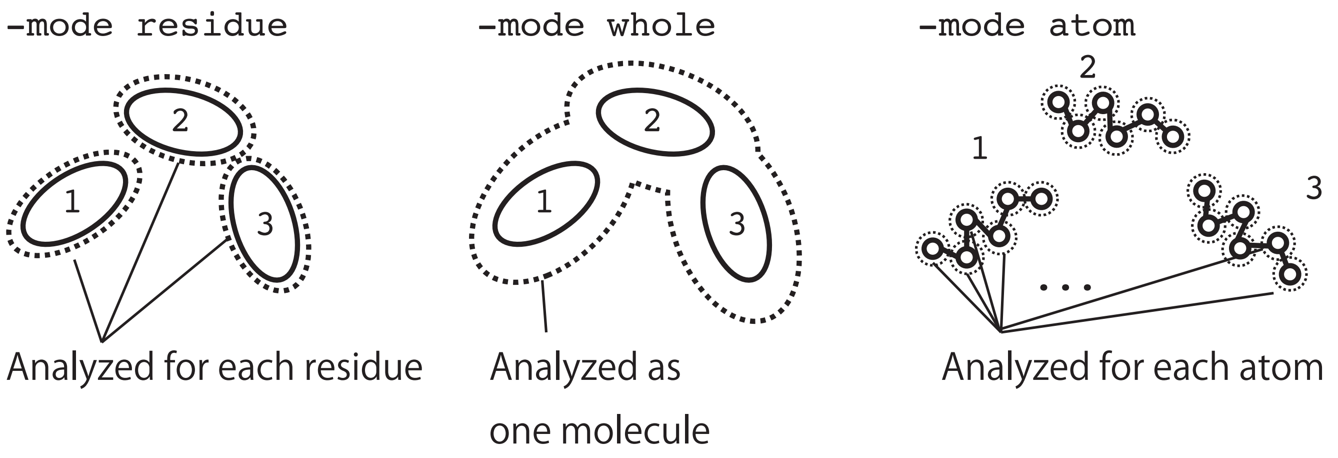

Fig. 3.1: Usage of the `-mode` option. In `residue`, the selected region is split by residue. In `whole`, the entire selection is treated as one molecule. In `atom`, each atom in the selection is treated as a separate molecule.

- **-fhead**
  - Default: Mode-dependent
  - Example: `-fhead out`

  Specifies the prefix for output filenames. For example, if set to `out`, output files like `out.XXX`, `out.YYY` will be generated.

- **-prep_only**
  - Default: `false`
  - Example: `-prep_only true`

  Specifies whether to stop after preprocessing (e.g., generating input parameter files) without running the analysis itself.

3.3 Trajectory Conversion (`trjconv`)

The `trjconv` mode performs various transformations on MD trajectories. It is typically used for the following purposes:

- To change the file format of a trajectory.
- To generate a trajectory containing only a specific region of interest
  (e.g., to remove water molecules from the trajectory).
- To apply periodic boundary wrapping (PBC wrapping) to an unwrapped trajectory.
- To fit the system such that a selected region aligns with a reference structure.
- To center the system such that a selected region is translated to the origin.

3.3.1 List of Options

Option	**Remarks**	**Option**	**Remarks**
`-stype`	Common	`-sfile`	Common
`-tintype`	Common	`-tin`	Common
`-totype`	Common	`-to`	Common
`-sel`x	Common	`-beg`	Mode specific
`-end`	Mode specific	`-fit`	Mode specific
`-centering`	Mode specific	`-wrap`	Mode specific
`-wrapcenter`	Mode specific	`-wrapcomp`	Mode specific
`-refpdb`	Mode specific	`-outselid`	Mode specific
`-fitselid`	Mode specific	`-refselid`	Mode specific

- **-beg <first frame to read>**
  - Default: `1`
  - Example: `-beg 10`

  Specifies the first frame to read from the trajectory. Note that frame indexing starts from 1. The example above starts reading from the 10th frame.

- **-end <last frame to read>**
  - Default: `0` (interpreted as the final frame)
  - Example: `-end 100`

  Specifies the last frame to read from the trajectory. Frame numbers are counted starting from 1. The example above reads up to the 100th frame.

- **-fit <true or false>**
  - Default: `false`
  - Example: `-fit true`
  - Note: If set to `true`, the options `-fitselid`, `-refpdb`, and `-refselid` must be specified.

  Enables or disables structural fitting. Fitting refers to the operation of translating and rotating the entire system so that a specific region matches a reference structure. When this option is used, the following must be specified:

  - `-fitselid`: selid of the fitting region in the trajectory
  - `-refpdb`: PDB file of the reference structure
  - `-refselid`: selid of the fitting region in the reference

- **-wrap <true or false>**
  - Default: `false`
  - Example: `-wrap true -wrapcomp fragment -wrapcenter origin`
  - Note: Use `-wrapcenter` to change the wrapping center from the origin.
    Use `-wrapcomp` to define the unit for wrapping.

  Specifies whether to apply periodic boundary wrapping (PBC) to the trajectory. By default, the wrapping center is the origin $(x, y, z) = (0, 0, 0)$, but it will be automatically changed to the center of mass of a region specified with `-wrapcenter`.

- **-centering <true or false>**
  - Default: `false`

  - Example: `-centering true -sel1 resid 1 to 200 -fitselid 1`
  - Note: Use `-fitselid` to specify the selid of the region to be centered.

  Specifies whether to perform centering of the system. Centering refers to translating the system so that the center of mass of a selected region is placed at the coordinate origin. The region to be centered is specified via `-fitselid`.

- **-refpdb <PDB file>**
  - Default: N/A
  - Example: `-refpdb A.pdb -sel2 resid 1 to 200 -refselid 2`
  - Note: Specify the selid of the fitting region in the reference using `-refselid`.

  Specifies the PDB file of the reference structure for fitting. This option is required if `-fit true` is specified. The fitting region within the reference is selected using `-refselid`.

- **-outselid <selid>**
  - Default: N/A
  - Example: `-sel0 resid 1 to 200 -outselid 0`

  Specifies the selid corresponding to the region to be written to the output trajectory.

- **-fitselid <selid>**
  - Default: N/A
  - Example: `-sel1 resid 1 to 200 -fitselid 1`

  Specifies the selid of the fitting region in the input trajectory.

- **-wrapcenter <origin or com>**
  - Default: `origin`
  - Example: `-wrapcenter origin`
  - Note: If `com` is specified, the selid must be provided using `-fitselid`.

  Specifies the center of the system used during wrapping.

  - **origin**: Use the coordinate origin as the center.
  - **com**: Use the center of mass of a selected region as the center.
    The selid of this region must be given with `-fitselid`.
    If fitting or centering is performed simultaneously, `com` is enforced.

- **-wrapcomp <fragment or residue or nocomp>**
  - Default: `fragment`
  - Example: `-wrapcomp fragment`

  Specifies the unit by which molecules are treated for PBC wrapping. For example, when atoms are partially outside the box, this determines whether to wrap individual atoms or entire molecular units.

  - **fragment**: Atoms connected through bonding paths are treated as a single unit.
  - **residue**: All atoms within a residue ID are treated as a single unit.
  - **nocomp**: Each atom is treated as an independent unit.

3.3.2 Examples of Analysis

Below are script examples using the minimum necessary options for common analyses. Options such as `-beg` and `-end` are omitted.

3.3.2.1 Removing Water from Trajectory

```
#!/bin/bash

anatra trjconv                          \
  -stype      parm7                     \
  -sfile      complex.prmtop            \
  -tintype    dcd                       \
  -tin        INP.dcd                   \
  -totype     dcd                       \
  -to         OUT.dcd                   \
  -sel0       not water                 \
  -outselid   0
```

3.3.2.2 PBC Wrapping

```
#!/bin/bash

anatra trjconv                          \
  -stype      parm7                     \
  -sfile      complex.prmtop            \
  -tintype    dcd                       \
```

(continues on next page)

(continued from previous page)

```
  -tin        INP.dcd                    \
  -totype     dcd                        \
  -to         OUT.dcd                    \
  -sel0       all                        \
  -wrap       true                       \
  -wrapcenter origin                     \
  -wrapcomp   fragment                   \
  -outselid   0
```

3.3.2.3 Fitting

This example assumes a system with one protein in aqueous solution (`resid 1 to 192`). Fitting is performed to align the Cα atoms in the trajectory to those in the reference structure:

```
anatra trjconv                           \
  -stype      parm7                      \
  -sfile      complex.prmtop             \
  -tintype    dcd                        \
  -tin        INP.dcd                    \
  -totype     dcd                        \
  -to         OUT.dcd                    \
  -refpdb     REF.pdb                    \
  -sel0       all                        \
  -sel1       resid 1 to 192 and name CA \
  -sel2       resid 1 to 192 and name CA \
  -fit        true                       \
  -outselid   0                          \
  -fitselid   1                          \
  -refselid   2
```

3.3.2.4 Centering

This example centers the protein′s center of mass at the origin:

```
anatra trjconv                           \
  -stype      parm7                      \
  -sfile      complex.prmtop             \
  -tintype    dcd                        \
  -tin        INP.dcd                    \
  -totype     dcd                        \
```

(continues on next page)

(continued from previous page)

```
  -to         OUT.dcd                \
  -sel0       all                    \
  -sel1       resid 1 to 192         \
  -centering  true                   \
  -outselid   0                      \
  -fitselid   1
```

3.3.2.5 PBC Wrapping and Fitting

This example performs both fitting and PBC wrapping. The wrapping center is set to the center of mass of the Cα atoms:

```
anatra trjconv                       \
  -stype      parm7                  \
  -sfile      complex.prmtop         \
  -tintype    dcd                    \
  -tin        INP.dcd                \
  -totype     dcd                    \
  -to         OUT.dcd                \
  -refpdb     REF.pdb                \
  -sel0       all                    \
  -sel1       resid 1 to 192 and name CA \
  -sel2       resid 1 to 192 and name CA \
  -fit        true                   \
  -wrap       true                   \
  -wrapcenter com                    \
  -wrapcomp   fragment               \
  -outselid   0                      \
  -fitselid   1                      \
  -refselid   2
```

3.3.2.6 PBC Wrapping and Centering

This example sets the wrapping and centering region to a lipid membrane (`resid 1 to 200 and segid MEMB`):

```
anatra trjconv                          \
  -stype      psf                       \
  -sfile      complex.psf               \
  -tintype    dcd                       \
  -tin        INP.dcd                   \
  -totype     dcd                       \
```

(continues on next page)

(continued from previous page)

```
-to         OUT.dcd                      \
-sel0       all                          \
-sel1       resid 1 to 200 and segid MEMB \
-centering  false                        \
-wrap       true                         \
-wrapcenter com                          \
-wrapcomp   fragment                     \
-outselid   0                            \
-fitselid   1
```

3.4 Distance Calculation (`distance`, `dis`)

The `distance` mode computes the distance between two selected regions. By default, it calculates the distance between the centers of mass of the selected molecules. It can also compute the minimum distance between atoms in two regions.

3.4.1 List of Options

Option	Remarks	Option	Remarks
`-stype`	Common	`-sfile`	Common
`-tintype`	Common	`-tin`	Common
`-flist_traj`	Common	`-fhead`	Common
`-sel0`	Common	`-sel1`	Common
`-mode0`	Common	`-mode1`	Common
`-pbc`	Mode specific	`-dt`	Mode specific
`-distance_type`†	Mode specific	`-mindist_type0`†	Mode specific
`-mindist_type1`†	Mode specific	`-prep_only`	Common

Options marked with † are not required for standard distance calculations and can be omitted in typical use cases.

- **-sel0 <selection>**
- **-sel1 <selection>**
  - Default: N/A
  - Example: `-sel0 resid 1 to 20 and noh -sel1 resname LIG`

  Specifies the two regions (selid = 0, 1) between which distances will be calculated.

- **-mode0 <residue or whole or atom>**
- **-mode1 <residue or whole or atom>**
  - Default: `residue`
  - Example: `-mode0 whole -mode1 residue`

  Defines how the selected regions (`-sel0`, `-sel1`) are partitioned for distance calculations.

- **-pbc <true or false>**
  - Default: `false`
  - Example: `-pbc true`

  Specifies whether periodic boundary conditions should be applied during distance calculations.

- **-dt <time interval>**
  - Default: `1.0`
  - Example: `-dt 0.1`

  Sets the time interval (first column of the output) for the time series data.

- **-distance_type <standard or minimum or intra>**†
  - Default: `standard`
  - Example: `-distance_type standard`

  Specifies the type of distance to compute:

  - **standard**: Standard distance between molecules.
  - **minimum**: Minimum atomic distance between regions.
  - **intra**: Intra-molecular distances.

  The `intra` option is useful when evaluating distances between atoms within the same molecule. For instance, in a system with 100 identical polymers (resname `PLM`) having terminal atoms named `CP1` and `CP2`, computing the end-to-end distance within each polymer requires care. Using `-distance_type standard` with `-sel0 name CP1` and `-sel1 name CP2` would also include inter-polymer distances. In contrast, `-distance_type intra` restricts calculations to within individual molecules.

- **-mindist_type1 <site or com>**†
  - Default: `site`
  - Example: `-mindist_type0 com -mindist_type1 site`

Specifies the candidate sites for minimum distance calculations:

- **site**: All atoms in the region are considered.
- **com**: Only the center of mass is considered.

When both regions use `com`, the result is equivalent to standard center-of-mass distance.

3.4.2 Output File Format

- `*.dis` file

  This file contains distances (or minimum distances) between the selected regions. Suppose `-mode0` and `-mode1` divide selections into n_1 and n_2 molecules, respectively. Then $n_1 \times n_2$ distances will be recorded per frame. The format is as follows:

  - Column 1: Time
  - Column 2: Distance between molecule pair (1, 1)
  - Column 3: Distance between pair (1, 2)
    …
  - Column n_2: Distance between pair (1, n_2)
  - Column $n_2 + 1$: Distance between pair (2, 1)
  - Column $n_2 + 2$: Distance between pair (2, 2)
    …

  All distances are reported in Å.

3.4.3 Example of Analysis

3.4.3.1 Standard Atom-to-Region Distance

The following example computes distances between a protein (`resid 1 to 192`) and 10 ligand molecules (`resname LIG`):

```
#!/bin/bash

anatra distance                                    \
  -stype          parm7                            \
  -sfile          complex.prmtop                   \
  -tintype        dcd                              \
  -tin            INP.dcd                          \
  -fhead          out                              \
  -pbc            true                             \
```

(continues on next page)

(continued from previous page)

```
-distance_type  standard                \
-dt             0.1                     \
-sel0           resid 1 to 192          \
-sel1           resname LIG             \
-mode0          whole                   \
-mode1          residue
```

3.5 Center of Mass Analysis and Mean Square Displacement (center_of_mass, com)

The program `center_of_mass` (`com`) computes the temporal variation of molecular centers of mass and the mean square displacement (MSD). To support two-dimensional diffusion (lateral diffusion) within lipid membranes, it also allows for MSD calculations restricted to the xy-plane.

3.5.1 Theoretical Background

3.5.1.1 Mean Square Displacement

Here, we introduce the definition of the mean square displacement (MSD) $\chi(t)$ and discuss its significance based on the diffusion equation.

Let $\mathbf{r}(t)$ denote the position of a molecule at time t. The MSD is defined as follows:

$$\chi\left(t\right) = \left\langle \left|\mathbf{r}\left(t\right) - \mathbf{r}\left(0\right)\right|^2 \right\rangle . \tag{3.1}$$

That is, $\chi(t)$ represents the time-averaged squared displacement of a molecule after a time interval t. To understand how MSD is described by the diffusion equation, let us consider a probability density $\rho(\mathbf{r}, t)$ representing the location of a molecule, and a diffusion coefficient D. The diffusion equation is written as

$$\frac{\partial}{\partial t}\rho\left(\mathbf{r}, t\right) = D\nabla^2\rho\left(\mathbf{r}, t\right). \tag{3.2}$$

Assuming that the molecule was initially located at $\mathbf{0}$ at time $t = 0$, the initial condition is given by

$$\rho\left(\mathbf{r}, t = 0\right) = \delta\left(\mathbf{r}\right) \tag{3.3}$$

Solving the diffusion equation under this condition yields

$$\rho\left(\mathbf{r}, t\right) = \frac{1}{\left(4\pi D t\right)^{3/2}} \exp\left[-\frac{\left|\mathbf{r}\right|^2}{4Dt}\right]. \tag{3.4}$$

Using this solution, the MSD can be expressed as

$$\chi\left(t\right) = \int d\mathbf{r}\ \left|\mathbf{r}\right|^2 \rho\left(\mathbf{r}, t\right) \tag{3.5}$$

$$= 6Dt, \tag{3.6}$$

indicating that MSD increases linearly with time, with a slope of $6D$. Because the diffusion equation is valid only in the long-time limit, the diffusion coefficient should be defined as

$$D = \frac{1}{6} \lim_{t \to \infty} \frac{\langle |\mathbf{r}(t) - \mathbf{r}(0)|^2 \rangle}{t}, \tag{3.7}$$

This expression is known as Einstein′s relation. It indicates that the diffusion coefficient can be computed from the slope of the MSD in the long-time regime.

3.5.2 List of Options

Option	Description	Option	Description
`-stype`	Common	`-sfile`	Common
`-tintype`	Common	`-tin`	Common
`-flist_traj`	Common	`-fhead`	Common
`-sel0`	Common	`-mode`	Common
`-outmsd`	Mode specific	`-unwarp`	Mode specific
`-msddim`	Mode specific	`-dt`	Mode specific
`-t_sparse`	Mode specific	`-t_range`	Mode specific
`-prep_only`	Common		

- **-outmsd <true or false>**
  - Default: `false`
  - Example: `-outmsd true`
  - Note: If set to `true`, both `-msddim` and `-msdrange` must also be specified.

  Specifies whether or not to compute the MSD.

- **-unwrap <true or false>**
  - Default: `false`
  - Example: `-unwrap false`

  Specifies whether to unwrap the trajectory. Since wrapped trajectories cannot yield accurate MSD values, it is necessary to set this option to `true`. Note that trajectories segmented by periodic boundary conditions (PBC) — as in GROMACS — are not supported by this option. In such cases, preprocessing with GROMACS `trjconv` is required.

- **-msddim <2 or 3>**
  - Default: `3`

  - Example: `-msddim 3`

  Specifies the dimensionality of the MSD calculation. Use `3` for general diffusion analysis and `2` for lateral diffusion within lipid bilayers.

- **-dt**
  - Default: `0.1`
  - Example: `-dt 0.1`

  Specifies the time interval between snapshots.

- **-t_range**
  - Default: `10`
  - Example: `-t_range 10`

  Specifies the time scale over which MSD is computed.

- **-t_sparse**

  Specifies the time interval for MSD output. For example, with `-dt 0.1` ($= \delta t$) and `-t_sparse 1` ($= \Delta t$), the MSD (in Å^2) is computed at time intervals:

$$\Delta T = \mathrm{nint}\left(\frac{\Delta t}{\delta t}\right)\delta t \tag{3.8}$$

3.5.3 Output File Formats

- `*.com`

  This file contains the center-of-mass coordinates of each molecule within the selected region, in `xyz` format.

- `*.msd`

  This file stores the MSD values for individual molecules. Output is generated only when `-outmsd true` is specified.

  - Column 1: Time
  - Column 2: MSD of molecule 1
  - Column 3: MSD of molecule 2
    …

- `*.msdave`

  This file contains the averaged MSD across all molecules. Output is generated only when `-outmsd true` is specified.

  - Column 1: Time

- Column 2: Mean MSD
- Column 3: Standard deviation of MSD

3.5.4 Examples of Analysis

3.5.4.1 MSD Calculation for a Protein

The following is an example of calculating the MSD of a single protein (`resid 1 to 192`) dissolved in aqueous solution. Since the protein consists of multiple amino acid residues, it is necessary to specify the mode as “`-mode whole`”.

```
#!/bin/bash

anatra center_of_mass        \
  -stype     parm7           \
  -sfile     complex.prmtop  \
  -tintype   dcd             \
  -tin       INP.dcd         \
  -fhead     out             \
  -outmsd    true            \
  -sel0      resid 1 to 192  \
  -mode      whole           \
  -msddim    3               \
  -dt        0.01            \
  -t_range   100             \
  -t_sparse  0.1
```

3.5.4.2 MSD Calculation for Multiple Small Molecules

Consider a system containing 100 ligand molecules (`resname LIG`) dissolved in aqueous solution. The following example computes both the MSD of each molecule and their average. In this case, molecular boundaries can be determined based on residue numbers, so the mode is set to “`-mode residue`”.

```
#!/bin/bash

anatra center_of_mass        \
  -stype     parm7           \
  -sfile     complex.prmtop  \
  -tintype   dcd             \
  -tin       INP.dcd         \
  -fhead     out             \
  -outmsd    true            \
```

(continues on next page)

(continued from previous page)

```
-sel0      resname LIG     \
-mode      residue         \
-msddim    3               \
-dt        0.01            \
-t_range   100             \
-t_sparse  0.01
```

3.6 Radial Distribution Function (radial_distr, rdf)

3.6.1 Theoretical Background

This section explains the definition of the radial distribution function (RDF).

3.6.1.1 RDF for Identical Particles

Consider a system containing N_a particles of type a, with a total volume V and a bulk number density $\rho_a = N_a/V$. The local density field $\hat{\rho}_a(\mathbf{r})$ is defined as

$$\hat{\rho}_a(\mathbf{r}) = \sum_{i=1}^{N_a} \delta(\mathbf{r} - \mathbf{r}_i). \tag{3.9}$$

Here, $\mathbf{r}_i$ is the position of the i-th particle of type a. If no external field is present, the ensemble average of the local density field is reduced to the bulk density.

$$\langle \hat{\rho}_a(\mathbf{r}) \rangle = \rho_a. \tag{3.10}$$

The, let us define the two-body density correlation function as

$$\chi(\mathbf{r}, \mathbf{r}') = \langle \hat{\rho}_a(\mathbf{r}) \hat{\rho}_a(\mathbf{r}') \rangle = \left\langle \sum_{i=1}^{N_a} \sum_{j=1}^{N_a} \delta(\mathbf{r} - \mathbf{r}_i) \delta(\mathbf{r}' - \mathbf{r}_j) \right\rangle. \tag{3.11}$$

Separating the terms for $i = j$ and $i \neq j$ gives

$$\chi(\mathbf{r}, \mathbf{r}') = \sum_{i=1}^{N_a} \langle \delta(\mathbf{r} - \mathbf{r}_i) \delta(\mathbf{r}' - \mathbf{r}_i) \rangle + \sum_{i=1}^{N_a} \sum_{j \neq i}^{N_a} \langle \delta(\mathbf{r} - \mathbf{r}_i) \delta(\mathbf{r}' - \mathbf{r}_j) \rangle. \tag{3.12}$$

The first term represents the probability that the same particle exists at both $\mathbf{r}$ and $\mathbf{r}'$, which is clearly zero unless $\mathbf{r} = \mathbf{r}'$, i.e.,

$$\sum_{i=1}^{N_a} \langle \delta(\mathbf{r} - \mathbf{r}_i) \delta(\mathbf{r}' - \mathbf{r}_i) \rangle = \rho_a \delta(\mathbf{r} - \mathbf{r}'). \tag{3.13}$$

The second term, called the two-body density, represents the probability that one particle is at $\mathbf{r}$ and another is at $\mathbf{r}'$:

$$\rho_{aa}^{(2)}(\mathbf{r}, \mathbf{r}') = \sum_{i=1}^{N_a} \sum_{j \neq i}^{N_a} \langle \delta(\mathbf{r} - \mathbf{r}_i) \delta(\mathbf{r}' - \mathbf{r}_j) \rangle. \tag{3.14}$$

If $\mathbf{r}$ and $\mathbf{r}'$ are far apart, their correlation is negligible. Thus

$$\rho_{aa}^{(2)}\left(\mathbf{r},\mathbf{r}'\right) \approx \rho_a^2\left(1-\frac{1}{N_a}\right). \qquad (|\mathbf{r}-\mathbf{r}'|\to\infty) \tag{3.15}$$

The pair distribution function is defined as

$$g_{aa}^{(2)}\left(\mathbf{r},\mathbf{r}'\right) = \frac{1}{\rho_a^2(1-1/N_a)}\rho_{aa}^{(2)}\left(\mathbf{r},\mathbf{r}'\right). \tag{3.16}$$

For large systems, $1/N_a \approx 0$. Therefore,

$$g_{aa}^{(2)}\left(\mathbf{r},\mathbf{r}'\right) = \frac{1}{\rho_a^2}\rho_{aa}^{(2)}\left(\mathbf{r},\mathbf{r}'\right). \tag{3.17}$$

In uniform systems, the function depends only on the relative position, indicating

$$g_{aa}^{(2)}\left(\mathbf{r},\mathbf{r}'\right) = g_{aa}^{(2)}\left(\mathbf{0},\mathbf{r}'-\mathbf{r}\right). \tag{3.18}$$

Let $\mathbf{r}$ be $\mathbf{r}'-\mathbf{r}$ and $g_{aa}\left(\mathbf{r}\right)$ be the spatial distribution function (SDF) defined as

$$g_{aa}\left(\mathbf{r}\right) = g_{aa}^{(2)}\left(\mathbf{0},\mathbf{r}\right). \tag{3.19}$$

Averaging over orientation leads to the radial distribution function (RDF) defined as

$$g_{aa}\left(r\right) = \frac{1}{4\pi r^2}\int d\mathbf{r}\,\delta\left(|\mathbf{r}|-r\right)g_{aa}\left(\mathbf{r}\right). \tag{3.20}$$

For isotropic systems, the following relationship holds.

$$g_{aa}\left(|\mathbf{r}|\right) = g_{aa}(\mathbf{r}). \tag{3.21}$$

RDF gives the relative probability of finding a particle at distance r compared to bulk. $g_{aa}(r) > 1$ indicates higher probability, $g_{aa}(r) < 1$ indicates lower probability than bulk.

3.6.1.2 RDF for Heterogeneous Particles

Consider a system with N_a particles of type a and N_b of type b, with respective bulk densities $\rho_a = N_a/V$ and $\rho_b = N_b/V$. The two-body density is

$$\rho_{ab}^{(2)}(\mathbf{r},\mathbf{r}') = \sum_{i=1}^{N_a}\sum_{j=1}^{N_b}\left\langle\delta(\mathbf{r}-\mathbf{r}_i^a)\delta(\mathbf{r}'-\mathbf{r}_j^b)\right\rangle. \tag{3.22}$$

For uncorrelated particles,

$$\rho_{ab}^{(2)}(\mathbf{r},\mathbf{r}') \approx \rho_a\rho_b. \tag{3.23}$$

The spatial distribution function is

$$g_{ab}(\mathbf{r}) = \frac{1}{\rho_a\rho_b}\rho_{ab}^{(2)}(\mathbf{0},\mathbf{r}), \tag{3.24}$$

and the RDF is

$$g_{ab}(r) = \frac{1}{4\pi r^2}\int d\mathbf{r}\,\delta(|\mathbf{r}|-r)g_{ab}(\mathbf{r}). \tag{3.25}$$

3.6.2 Option List

Option	Description	Option	Description
`-stype`	Common	`-sfile`	Common
`-tintype`	Common	`-tin`	Common
`-flist_traj`	Common	`-fhead`	Common
`-sel0`	Common	`-sel1`	Common
`-mode0`	Common	`-mode1`	Common
`-dr`	Mode specific	`-identical`	Mode specific
`-normalize`	Mode specific	`-separate_self`	Mode specific

- **-dr**
  - Default: `0.1`

  Specifies the bin width for RDF calculation.

- **-identical**
  - Default: `false`

  Set to `true` when `-sel0` and `-sel1` are the same. As discussed in the theory section, RDF calculations differ depending on whether the particles are identical or not, so this must be specified by the user.

- **-normalize**
  - Default: `true`

  Specifies whether to output the RDF $g_{12}(r)$ directly (`true`), or multiply by the average number density ρ_2 of `sel1` to output $\rho_{12}(r) = \rho_2 g_{12}(r)$ (`false`).

- **-separate_self**
  - Default: `false`

  Determines whether to separate intra- and intermolecular contributions in RDF. This is useful, for example, when calculating RDF between different atoms in the same type of molecule (e.g., OW-HW1 in water). Setting `true` separates contributions from within the same molecule.

3.6.3 Output File Format

- `*.rdf` file

  This file contains the RDF output. The content varies depending on the `-separate_self` option:

  - If `-separate_self false`:
    - Column 1: Distance
    - Column 2: RDF
  - If `-separate_self true`:
    - Column 1: Distance
    - Column 2: Intermolecular RDF
    - Column 3: Intramolecular RDF (self-correlation function)

3.6.4 Examples of Analysis

3.6.4.1 RDF between oxygen atoms of water molecules

```
#!/bin/bash

anatra rdf                                        \
  -stype         parm7                            \
  -sfile         complex.prmtop                   \
  -tintype       dcd                              \
  -tin           INP.dcd                          \
  -fhead         out                              \
  -sel0          resname WAT and name O           \
  -sel1          resname WAT and name O           \
  -mode0         residue                          \
  -mode1         residue                          \
  -dr            0.1                              \
  -identical     true                             \
  -normalize     true                             \
  -separate_self false
```

3.6.4.2 RDF between oxygen and hydrogen atoms in water molecules

```
#!/bin/bash

anatra rdf                                       \
  -stype         parm7                           \
  -sfile         complex.prmtop                  \
  -tintype       dcd                             \
  -tin           INP.dcd                         \
  -fhead         out                             \
  -sel0          resname WAT and name O          \
  -sel1          resname WAT and name H1         \
  -mode0         residue                         \
  -mode1         residue                         \
  -dr            0.1                             \
  -identical     false                           \
  -normalize     true                            \
  -separate_self true
```

3.6.4.3 RDF between protein Cα atoms and water oxygen atoms

This example calculates the RDF between the Cα atoms (`resid 1 to 192`) of a protein and oxygen atoms of water molecules, and outputs the average of the individual RDFs.

```
anatra rdf                                       \
  -stype         parm7                           \
  -sfile         complex.prmtop                  \
  -tintype       dcd                             \
  -tin           INP.dcd                         \
  -fhead         out                             \
  -sel0          resid 1 to 192 and name CA      \
  -sel1          resname WAT and name O          \
  -mode0         atom                            \
  -mode1         residue                         \
  -dr            0.1                             \
  -identical     false                           \
  -normalize     true                            \
  -separate_self false
```

3.7 Spatial Distribution Function (`spatial_distr`, `sdf`)

Note: The `spatial_distr` mode is not available when compiled with the gcc compiler.

3.7.1 Theoretical Background

Consider a system with N_a particles of type a, volume V, and number density $\rho_a = N_a/V$. Assume that one molecule is fixed at the center of the system, including its orientation, acting effectively as an external field. Although such a situation does not naturally occur in standard solution-phase MD simulations, a virtually equivalent condition can be constructed by aligning all trajectories to a reference structure (fitting), as described at the end of this section.

Similar to the radial distribution function, we define the local number density of particle a as

$$\hat{\rho}_a(\mathbf{r}) = \sum_{i=1}^{N_a} \delta(\mathbf{r} - \mathbf{r}_i). \tag{3.26}$$

Its ensemble average under the external field gives the one-body density

$$\rho_a(\mathbf{r}) = \langle \hat{\rho}_a(\mathbf{r}) \rangle_{\text{ext}} = \left\langle \sum_{i=1}^{N_a} \delta(\mathbf{r} - \mathbf{r}_i) \right\rangle_{\text{ext}}. \tag{3.27}$$

Here, $\langle \cdots \rangle_{\text{ext}}$ denotes the ensemble average under the external field. In the absence of an external field or far from the fixed molecule, $\rho_a(\mathbf{r}) = \rho_a$. THen, we define the spatial distribution function (SDF) as:

$$g_a(\mathbf{r}) = \frac{\rho_a(\mathbf{r})}{\rho_a}. \tag{3.28}$$

If $g_a(\mathbf{r}) > 1$, the local probability of finding particle a is larger than the bulk value; if $g_a(\mathbf{r}) < 1$, it is lower than in bulk.

To compute the SDF, the system is first fitted such that the molecule treated as the external field overlaps with a reference structure. For unwrapped trajectories, PBC wrapping must be performed in advance (Fig. 3.2). The `spatial_distr` mode handles all these steps internally. Once transformed, the reference molecule is fixed in position and orientation, making it effectively act as a static field, and the surrounding molecular distribution can be analyzed as an SDF. If the reference molecule lacks a stable structure, fitting becomes unreliable and SDF computation is infeasible. However, for proteins with stable native structures and minimal conformational changes, SDF analysis is applicable.

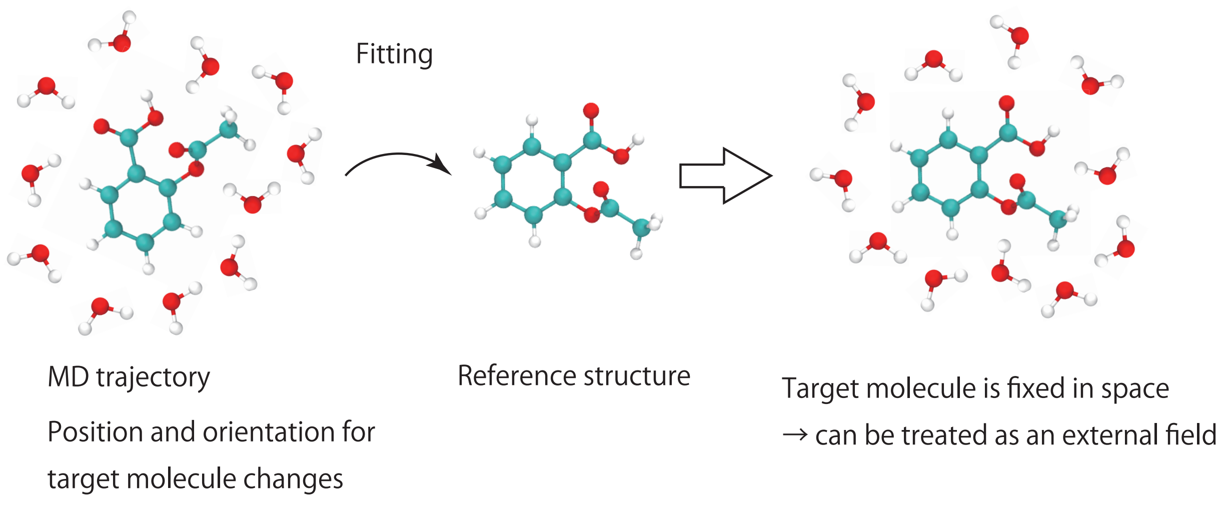
Fitting
MD trajectory
Position and orientation for target molecule changes
Reference structure
Target molecule is fixed in space
→ can be treated as an external field

Fig. 3.2: By aligning the system such that the molecule of interest overlaps with the reference structure (fitting), it becomes possible to examine the three-dimensional distribution of surrounding molecules relative to a fixed molecular orientation. The `spatial_distr` mode automates this entire procedure.

3.7.2 List of Options

Option	Remarks	Option	Remarks
`-stype`	Common	`-sfile`	Common
`-tintype`	Common	`-tin`	Common
`-flist_traj`	Common	`-fhead`	Common
`-selX`	Common	`-mode`	Common
`-ng3`	Mode specific	`-del`	Mode specific
`-origin`	Mode specific	`-fit`	Mode specific
`-refpdb`	Mode specific	`-refselid`	Mode specific
`-fitselid`	Mode specific	`-use_spline`	Mode specific
`-spline_resolution`	Mode specific	`-prep_only`	Common

Advanced options for conditional SDF analysis exist but will be officially supported in a future update.

- **-ng3 <number of grid points along x, y, z>**
  - Default: `50 50 50`
  - Example: `-ng3 50 50 50`

  Specifies the number of grid points along each spatial axis, n_x, n_y, n_z. The total number of grid points is $n_x n_y n_z$.

- **-del <grid spacing along x, y, z axes (Å)>**
  - Default: `0.5 0.5 0.5`
  - Example: `-del 0.5 0.5 0.5`

  Specifies the grid spacing $\Delta x, \Delta y, \Delta z$ in Å. The volume element per grid point is $\Delta V = \Delta x \Delta y \Delta z$.

- **-origin <coordinates of grid origin (x, y, z) in Å>**
  - Default: `0.0 0.0 0.0`
  - Example: `-origin 0.0 0.0 0.0`

  Sets the origin of the grid. For origin (x_0, y_0, z_0), the coordinates of grid point (i_x, i_y, i_z) are:

$$x_i = x_0 + \Delta x \,(i_x - 1) \tag{3.29}$$
$$y_i = y_0 + \Delta y \,(i_y - 1) \tag{3.30}$$
$$z_i = z_0 + \Delta z \,(i_z - 1) \tag{3.31}$$

- **-fit <true or false>**
  - Default: `false`
  - Example: `-fit true`

  If set to `true`, the system is PBC-wrapped and then fitted so that the solute molecule (treated as the external field) aligns with the reference structure. The following must be specified:

  - `-refpdb`
  - `-refselid`
  - `-fitselid`
  - `-sel0` (target for SDF calculation)
  - `-sel1` (reference fitting region)
  - `-sel2` (trajectory fitting region)

  Important: If the input trajectory has already been fitted, you must explicitly set `-fit false`. Applying PBC wrapping after fitting leads to invalid configurations.

- **-refselid <selid>**
  - Default: N/A
  - Example: `-refselid 1`

  Specifies the selid of the reference region used for fitting in the reference structure. For example, if using `-sel1` as the reference selection, set `-refselid 1`.

- **-fitselid <selid for fitting region in trajectory>**
  - Default: N/A
  - Example: `-fitselid 2`

  Specifies the selid for the region used for fitting within the trajectory. For example, if using `-sel2` for trajectory fitting, set `-fitselid 2`.

- **-use_spline <true or false>**
  - Default: `false`
  - Example: `-use_spline true`
  - Note: The resolution is controlled by `-spline_resolution`.

  Applies spline interpolation (Akima spline) to the SDF grid.

- **-spline_resolution <integer resolution factor>**
  - Default: `4`
  - Example: `-spline_resolution 4`
  - Note: Active only if `-use_spline true` is specified.

  Specifies the subdivision factor n_r for each grid axis. Given the original grid spacing $\Delta\alpha$ ($\alpha = x, y, z$), the refined grid spacing becomes:

$$\Delta\alpha' = \frac{\Delta\alpha}{n_r} \tag{3.32}$$

  The total number of grid points changes from N to:

$$N' = n_r^3 \cdot N \tag{3.33}$$

3.7.3 Output File Format

- `*_out_g_original.dx`
  SDF output in DX format. No spline interpolation applied.
- `*_out_g_spline.dx`
  SDF output in DX format with spline interpolation. Generated only if `-use_spline true` is specified.

3.7.4 Examples of Analysis

3.7.4.1 SDF of Water Molecules Around a Protein

The following script computes the SDF of water molecule centers of mass around a protein (`resid 1 to 192`):

```
#!/bin/bash

anatra spatial_distr                        \
  -stype             parm7                  \
  -sfile             complex.prmtop         \
  -tintype           dcd                    \
  -tin               INP.dcd                \
  -fhead             out                    \
  -sel0              water                  \
  -sel1              resid 1 to 192 and noh \
  -sel2              resid 1 to 192 and noh \
  -mode              residue                \
  -ng3               50 50 50               \
  -del               0.4 0.4 0.4            \
  -origin            -10.0 -10.0 -10.0      \
  -fit               true                   \
  -refpdb            ref.pdb                \
  -refselid          1                      \
  -fitselid          2                      \
  -use_spline        true                   \
  -spline_resolution 4
```

3.8 Root-Mean-Square Deviation (`rmsd`)

3.8.1 Theoretical Background

This section explains the definition of the root-mean-square deviation (RMSD). RMSD quantitatively evaluates the structural deviation of a molecule or a specific region of interest from a reference structure at a given time during an MD simulation. It is commonly used to assess how much a structure deviates from, for example, an experimental crystal structure during the equilibration phase, and whether or not it relaxes over time.

Consider a molecule composed of N_{sel} atoms. Let $\mathbf{r}_i$ denote the position of the i-th atom, $\mathbf{r}_i^{\mathrm{ref}}$ the corresponding coordinate in the reference structure, and $\mathbf{r}_i(t)$ the position at time t. Then, the RMSD $\chi(t)$ is defined as:

$$\chi(t) = \sqrt{\frac{1}{N_{\mathrm{sel}}} \sum_{i=1}^{N_{\mathrm{sel}}} \left| \mathbf{r}_i(t) - \mathbf{r}_i^{\mathrm{ref}} \right|^2}. \tag{3.34}$$

3.8.2 List of Options

Option	Remarks	Option	Remarks
`-stype`	Common	`-sfile`	Common
`-tintype`	Common	`-tin`	Common
`-fout`	Common	`-sel0`	Common
`-sel1`	Common	`-sel2`	Common
`-sel3`	Common	`-fit`	Mode specific
`-refpdb`	Common	`-fitselid`	Mode specific
`-refselid`	Mode specific	`-rmsdselid`	Mode specific
`-rmsdrefselid`	Mode specific		

- **-fout <output file name>**
  - Default: `out.rmsd`
  - Example: `-fout out.rmsd`

  Specifies the filename for the output of RMSD results.

- **-refpdb <reference structure in PDB format>**
  - Default: N/A
  - Example: `-refpdb ref.pdb`

  Specifies the reference structure used for RMSD calculation.

- **-fit <true or false>**
  - Default: `false`
  - Example: `-fit true`
  - Note: If set to `true`, `-refselid` and `-fitselid` must be specified.

  Indicates whether to perform structural fitting prior to RMSD calculation.

- **-fitselid <selid>**
  - Default: N/A
  - Example: `-fitselid 1`
- **-refselid <selid>**

  - Default: N/A
  - Example: `-refselid 2`

  Specify the selection IDs of the fitting regions in the trajectory and the reference structure, respectively. These options are required when `-fit true` is set.

- **-rmsdselid <selid>**
  - Default: N/A
  - Example: `-rmsdselid 2`

  Specifies the selid corresponding to the region used in the trajectory for RMSD calculation.

- **-rmsdrefselid <selid>**
  - Default: N/A
  - Example: `-rmsdrefselid 3`

  Specifies the selid corresponding to the region in the reference structure for RMSD calculation.

3.8.3 Output File Format

- `*.rmsd` file

  This file contains the time evolution of RMSD values.

  - Column 1: Time
  - Column 2: RMSD

3.8.4 Examples of Analysis

3.8.4.1 RMSD from the Crystal Structure of a Protein in Solution

This example calculates the RMSD of the Cα atoms of a protein (`resid 1 to 161`) with respect to a crystal structure.

```
anatra rmsd                                    \
  -stype        parm7                          \
  -sfile        complex.prmtop                 \
  -tintype      dcd                            \
  -tin          INP.dcd                        \
  -fout         out.rmsd                       \
  -sel0         resid 1 to 161 and name CA     \
  -sel1         resid 1 to 161 and name CA     \
```

(continues on next page)

(continued from previous page)

```
  -sel2         resid 1 to 161 and name CA    \
  -sel3         resid 1 to 161 and name CA    \
  -fit          true                          \
  -refpdb       ref.pdb                       \
  -fitselid     0                             \
  -refselid     1                             \
  -rmsdselid    2                             \
  -rmsdrefselid 3
```

3.9 Interaction Energy (`interaction_energy`, `ene`)

Note: The `interaction_energy` mode is not available when compiled with the gcc compiler.

The `interaction_energy` (`ene`) mode allows the calculation of interaction energies between selected molecules, groups of molecules, or specific regions. Currently, **ANATRA** supports the following three types of interactions:

- **Electrostatic Interaction**
  Electrostatic interaction energy U_{elec} between molecules A and B can be computed using two schemes. The first is the bare interaction,

$$U_{\mathrm{elec}} = \sum_{i \in \mathrm{A}} \sum_{j \in \mathrm{B}} \frac{q_i q_j}{r_{ij}}, \tag{3.35}$$

  where i and j denote atomic indices in molecules A and B, respectively. The second scheme uses the Smooth Particle Mesh Ewald (SPME) method, which is the standard approach for accounting for long-range electrostatic interactions in MD simulations. If consistency with MD simulations is required, the SPME method should be employed.

- **Lennard-Jones (LJ) Interaction**
  The LJ interaction, which empirically describes van der Waals interactions, is given by

$$U_{\mathrm{LJ}} = \sum_{i \in \mathrm{A}} \sum_{j \in \mathrm{B}} 4\epsilon_{ij} \left[\left(\frac{\sigma_{ij}}{r_{ij}} \right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}} \right)^{6} \right]. \tag{3.36}$$

  Here, ϵ_{ij} and σ_{ij} are the LJ parameters between atom i in molecule A and atom j in molecule B.

- **Attractive LJ Interaction**
  The attractive component of the LJ interaction is defined as:

$$U_{\mathrm{attr}} = \sum_{i \in \mathrm{A}} \sum_{j \in \mathrm{B}} u_{\mathrm{attr},ij}, \tag{3.37}$$

$$u_{\mathrm{attr},ij} = \begin{cases} 4\epsilon_{ij} \left[\left(\frac{\sigma_{ij}}{r_{ij}} \right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}} \right)^{6} \right] & r_{ij} \geq 2^{1/6}\sigma, \\ -\epsilon_{ij} & r_{ij} < 2^{1/6}\sigma. \end{cases} \tag{3.38}$$

  This formulation removes the repulsive part of the LJ potential, eliminating the multivalued nature of energy with respect to distance (see Fig. 3.3). When constructing a PMF for molecular binding, using the attractive LJ interaction as a reaction coordinate can more clearly distinguish between bound and unbound states than the standard LJ potential.

In addition to the individual components above, the sum of electrostatic and LJ can also be calculated.

3.9.1 List of Options

Option	Remarks	Option	Remarks
`-stype`	Common	`-sfile`	Common
`-tintype`	Common	`-tin`	Common
`-flist_traj`	Common	`-fhead`	Common
`-sel0`	Common	`-sel1`	Common
`-mode0`	Common	`-mode1`	Common
`-parmformat`	Mode specific	`-fanaparm`	Mode specific
`-pbc`	Mode specific	`-calc_vdw`	Mode specific
`-calc_elec`	Mode specific	`-vdw_type`	Mode specific
`-elec_type`	Mode specific	`-dt`	Mode specific
`-rvdwcut`	Mode specific	`-relcut`	Mode specific
`-pme_alpha`	Mode specific	`-pme_grids`	Mode specific
`-pme_rigid`	Mode specific	`-pme_dual`	Mode specific
`-pme_fprm`	Mode specific	`-prep_only`	Common

- **-parmformat <prmtop or anaparm>**
  - Default: `prmtop`
  - Example: `-parmformat anaparm`
  - Note: When set to `anaparm`, `-fanaparm` must also be specified.

  Specifies the parameter file for atomic charges. When using `prmtop`, the file given by `-sfile` will be automatically used.

- **-fanaparm <filename>**
  - Default: N/A
  - Example: `-fanaparm complex.anaparm`

  Specifies the `anaparm` file containing LJ parameters and atomic charges. Required when `-parmformat anaparm` is used.

- **-pbc <true or false>**
  - Default: `false`
  - Example: `-pbc true`

Enables periodic boundary condition treatment, including image interactions from adjacent periodic boxes.

- **-calc_vdw <true or false>**
  - Default: `true`
  - Example: `-calc_vdw false`

Specifies whether to compute van der Waals (LJ) interactions.

- **-calc_elec <true or false>**
  - Default: `false`
  - Example: `-calc_elec true`

Specifies whether to compute electrostatic interactions.

- **-vdw_type <standard or attractive>**
  - Default: `standard`
  - Example: `-vdw_type attractive`

Specifies which LJ interaction function to use.
`standard` computes the standard LJ (Eq. \eqref{LJ}), and `attractive` computes the attractive LJ (Eq. \eqref{Uattr}).

- **-elec_type <bare or pme>**
  - Default: `bare`
  - Example: `-elec_type bare`

Specifies the method for electrostatic interaction calculation:
`bare` for Eq. \eqref{elec}, `pme` for the PME method.

- **-dt <time interval>**
  - Default: `0.1`
  - Example: `-dt 1.0`

Sets the time resolution (first column) of the time-series output.

- **-rvdwcut <cutoff distance (Å)>**
  - Default: `12.0`
  - Example: `-rvdwcut 20.0`

Cutoff distance for van der Waals interactions.

- **-relcut <cutoff distance (Å)>**
  - Default: `1.0e10`
  - Example: `-relcut 1.0e10`

Cutoff distance for electrostatic interactions.

- **-pme_alpha <PME screening parameter (Å^{-1})>**
  - Default: `0.35`
  - Example: `-pme_alpha 0.35`

Specifies the screening parameter used in PME. Only used when `-elec_type pme`.

- **-pme_grids <grid size in x, y, z>**
  - Default: `64 64 64`
  - Example: `-pme_grids 64 64 64`

Grid size n_x, n_y, n_z used in PME. Total number of grid points is $n_x n_y n_z$.

- **-pme_rigid <true or false>**
  - Default: `false`
  - Example: `-pme_rigid false`

If the region specified by `-sel0` is spatially fixed, set to `true` to reduce PME computational cost.

3.9.2 Output File Format

- `*.uij` file

This file contains pairwise interaction energies between selected regions. If selections 0 and 1 are divided into n_1 and n_2 molecules, respectively (via `-mode0` and `-mode1`), the output contains $n_1 n_2$ interaction terms:

  - Column 1: Time
  - Column 2: Interaction between pair (1, 1)
  - Column 3: Interaction between pair (1, 2)
    …

- Column n_2: Interaction between pair (1, n_2)
- Column $n_2 + 1$: Interaction between pair (2, 1)
  …

3.9.3 Examples of Analysis

The following are minimal examples for common types of interaction energy analysis.

3.9.3.1 Electrostatic + LJ Interaction Between Solute Molecules

This example computes the total of electrostatic and LJ interactions between a protein (`resid 1 to 161`) and ligand molecules (`resname LIG`), using the SPME method for electrostatics:

```
#!/bin/bash

anatra energy                                \
  -stype       parm7                         \
  -sfile       complex.prmtop                \
  -tintype     dcd                           \
  -tin         INP.dcd                       \
  -fhead       out                           \
  -parmformat  prmtop                        \
  -pbc         true                          \
  -calc_vdw    true                          \
  -calc_elec   true                          \
  -vdw_type    standard                      \
  -elec_type   pme                           \
  -mode0       whole                         \
  -mode1       residue                       \
  -dt          0.1                           \
  -rvdwcut     12.0                          \
  -relcut      12.0                          \
  -pme_alpha   0.35                          \
  -pme_grids   64 64 64                      \
  -pme_rigid   false                         \
  -sel0        resid 1 to 161                \
  -sel1        resname LIG                   \
  -prep_only   false
```

3.9.3.2 LJ Interaction Between Solute Molecules

This example computes only the LJ interactions between a protein and ligand molecules:

```
#!/bin/bash

anatra energy                                \
  -stype       parm7                         \
  -sfile       complex.prmtop                \
  -tintype     dcd                           \
  -tin         INP.dcd                       \
  -fhead       out                           \
  -parmformat  prmtop                        \
  -pbc         true                          \
  -calc_vdw    true                          \
  -calc_elec   false                         \
  -vdw_type    standard                      \
  -mode0       whole                         \
  -mode1       residue                       \
  -dt          0.1                           \
  -rvdwcut     12.0                          \
  -relcut      1.0e10                        \
  -sel0        resid 1 to 161                \
  -sel1        resname LIG                   \
  -prep_only   false
```

3.10 Lipid Order Parameter (`lipid_order`, `scd`)

The `lipid_order` (`scd`) mode allows for the calculation of the orientational order parameter S_{CD} associated with acyl chains of phospholipids. Using the membrane normal direction as the z-axis, the angle θ is defined between this axis and the vector connecting a specific carbon atom and its bonded hydrogen. The S_{CD} parameter is then defined as:

$$S_{\mathrm{CD}} = \frac{1}{2}\left\langle 3\cos^2\theta - 1 \right\rangle . \tag{3.39}$$

The range of S_{CD} is $0 \leq S_{\mathrm{CD}} \leq 1$, where higher values indicate greater order, and a value of 0 corresponds to complete disorder. Since S_{CD} is defined for each carbon atom, this analysis provides position-resolved information on lipid order along the acyl chains.

3.10.1 List of Options

Option	Remarks	Option	Remarks
`-stype`	Common	`-sfile`	Common
`-tintype`	Common	`-tin`	Common
`-flist_traj`	Common	`-fhead`	Common
`-selX`	Common	`-mode`	Common
`-cindex`	Mode specific	`-prep_only`	Common

- **-selX <selection of atoms>**
  - Default: N/A
  - Example: `-sel0 name C32 H2X H2Y`

  Specify the carbon atom and its bonded hydrogen atoms for which S_{CD} will be calculated. Use `-selX` for each carbon atom of interest.

- **-cindex <carbon indices>**
  - Default: N/A
  - Example: `-cindex 1 2 3 4 5`

  Specifies the carbon index number (counted from the base of the acyl chain) corresponding to each `-selX` entry. The number of values given must match the number of `-selX` options. The carbon indices provided here will be used for the first column (x-axis) of the output `.scd` file.

3.10.2 Output File Format

- `*.scd`
  The S_{CD} values are printed in the order of carbon atoms specified by `-sel0`, `-sel1`, $\cdots$
  - Column 1: Carbon index (as specified by `-cindex`)
  - Column 2: S_{CD}

3.10.3 Examples of Analysis

3.10.3.1 Calculation of S_{CD} in a DPPC Lipid Bilayer

```
anatra lipid_order                             \
  -stype      psf                              \
  -sfile      complex.psf                      \
  -tintype    netcdf                           \
  -flist_traj flist                            \
  -fhead      run_nc                           \
  -dt         0.1                              \
  -cindex     1 2 3 4 5                        \
  -sel0       name C23 H3R H3S and segid MEMB \
  -sel1       name C24 H4R H4S and segid MEMB \
  -sel2       name C25 H5X H5Y and segid MEMB \
  -sel3       name C26 H6X H6Y and segid MEMB \
  -sel4       name C27 H7X H7Y and segid MEMB \
  -prep_only  false
```

3.11 Distribution Function Along the Membrane Normal (z-axis) (z_profile, zprof)

The `z_profile` (`zprof`) mode calculates the distribution of molecular species along the membrane normal (i.e., the z-axis). The local density field of species a along the z-axis is defined as

$$\hat{\rho}_a(z) = \frac{1}{A_{xy}} \sum_{i=1}^{N_a} \delta(z - z_i), \tag{3.40}$$

where A_{xy} is the cross-sectional area of the simulation box in the xy-plane, and z_i is the z-coordinate of the i-th molecule. The ensemble-averaged distribution function is defined as

$$\rho_a(z) = \langle \hat{\rho}_a(z) \rangle . \tag{3.41}$$

For comparison with experimental data, it is often useful to compute the electron density distribution $\rho_a^e(z)$. The corresponding local density field is defined as

$$\hat{\rho}_a^e(z) = \sum_{i=1}^{N_a} \sum_{\lambda=1}^{N_s} q_\lambda \delta(z - z_{i,\lambda}), \tag{3.42}$$

where q_λ is the number of electrons on the λ-th atom in molecule a, and $z_{i,\lambda}$ is its z-coordinate in the i-th molecule. The ensemble average yields the electron density distribution: $\rho_a^e(z) = \langle \hat{\rho}_a^e(z) \rangle$.

3.11.1 List of Options

Option	Remarks	Option	Remarks
`-stype`	Common	`-sfile`	Common
`-tintype`	Common	`-tin`	Common
`-flist_traj`	Common	`-fhead`	Common
`-sel0`	Common	`-sel1`	Common
`-mode0`	Common	`-mode1`	Common
`-target_selid`	Mode specific	`-center_selid`	Mode specific
`-denstype`	Mode specific	`-symmetrize`	Mode specific
`-prep_only`	Common	–	–

- **-sel0 <selection>**
- **-sel1 <selection>**
  - Default: N/A
  - Example: `-sel0 resname ETOH -sel1 segid MEMB`

  Select the molecules for which the distribution function is to be calculated. To define the membrane center as $z = 0$, membrane molecules must also be selected.

- **-target_selid <selid>**
  - Default: N/A
  - Example: `-target_selid 0`

  Specifies the selid (defined via `-selX`) corresponding to the molecules for which the distribution is to be calculated.

- **-center_selid <selid>**
  - Default: N/A
  - Example: `-center_selid 1`

  Specifies the selid (defined via `-selX`) corresponding to the membrane molecules whose center-of-mass is used to define $z = 0$.

- **-symmetrize <true or false>**
  - Default: `false`
  - Example: `-symmetrize false`

Specifies whether to symmetrize the distribution by averaging the data for $z > 0$ and $z < 0$.

- **-denstype <number or electron>**
  - Default: `number`
  - Example: `-denstype number`

  Specifies the type of density to compute:
  `number` for number density, or `electron` for electron density.

3.11.2 Output File Format

- `*.zprof`

  The distribution function of the molecules corresponding to the `-target_selid` will be output in the following format:
  - Column 1: z-coordinate
  - Column 2: Density value

3.11.3 Examples of Analysis

3.11.3.1 Number Density Profile of Water in a Lipid Membrane System

```
#!/usr/bin/env bash

anatra z_profile                 \
  -stype      psf                \
  -sfile      complex.psf        \
  -tintype    netcdf             \
  -flist_traj flist              \
  -fhead      run                \
  -sel0       water              \
  -sel1       segid MEMB and noh \
  -mode0      residue            \
  -mode1      whole              \
  -target_selid 0                \
  -center_selid 1                \
  -denstype   number             \
  -prep_only  false
```

3.12 Molecular Orientation Along a Specific Direction (`z_orient`, `zori`)

The `z_orient` (`zori`) mode computes the orientation distribution of molecules relative to a reference direction, typically the z-axis.

To define the orientation of a molecule, two parts of the molecule are selected: the *bottom* and the *head*. The centers of mass of these two parts are denoted by $\mathbf{R}_\mathrm{B}$ and $\mathbf{R}_\mathrm{H}$, respectively. The molecular axis $\mathbf{u}$ is then defined as

$$\mathbf{u} = \mathbf{R}_\mathrm{H} - \mathbf{R}_\mathrm{B}. \tag{3.43}$$

Let θ be the angle between the z-axis (unit vector $\mathbf{e}_z$) and the molecular axis (unit vector $\mathbf{e}_\mathrm{u}$). Then

$$\cos\theta = \mathbf{e}_z \cdot \mathbf{e}_\mathrm{u}. \tag{3.44}$$

The orientation distribution function is defined by

$$P(\chi) = C \left\langle \delta(\chi - \cos\theta) \right\rangle , \tag{3.45}$$

where C is a normalization constant determined such that

$$\int_0^\pi d\theta \, \sin\theta \, P(\chi) = 1. \tag{3.46}$$

By changing variables from θ to $\chi = \cos\theta$, the left hand side of the above equation becomes

$$\begin{aligned} \int_0^\pi d\theta \, \sin\theta \, P(\chi) &= \int_0^1 d\chi \, P(\chi) \\ &= CN \end{aligned} \tag{3.47}$$

where N is the number of targer molecules. Thus, C is set to $C = 1/N$. The `z_orient` mode calculates $P(\chi) = P(\cos\theta)$.

Additionally, the reference z-axis does not have to be fixed in the laboratory frame; it can also be defined by the orientation of a particular molecule. If four points $\mathbf{x}_{\mathrm{B}_0}, \mathbf{x}_{\mathrm{H}_0}, \mathbf{x}_{\mathrm{B}_1}, \mathbf{x}_{\mathrm{H}_1}$ are chosen from this reference molecule, the two vectors:

$$\mathbf{v}_0 = \mathbf{x}_{\mathrm{H}_0} - \mathbf{x}_{\mathrm{B}_0}, \quad \mathbf{v}_1 = \mathbf{x}_{\mathrm{H}_1} - \mathbf{x}_{\mathrm{B}_1} \tag{3.48}$$

can be used to define the z-axis unit vector via the cross product:

$$\mathbf{e}_z = \frac{\mathbf{v}_1 \times \mathbf{v}_2}{|\mathbf{v}_1 \times \mathbf{v}_2|}. \tag{3.49}$$

The orientation distribution $P(\cos\theta)$ is then computed based on the angle between this $\mathbf{e}_z$ and the molecular axis $\mathbf{u}$.

3.12.1 List of Options

Option	Remarks	Option	Remarks
`-stype`	Common	`-sfile`	Common
`-tintype`	Common	`-tin`	Common
`-flist_traj`	Common	`-fhead`	Common
`-sel`x	Common	`-mode`x	Common
`-zdef_type`	Mode specific	`-vecX_bottom_selid`	Mode specific
`-vecX_head_selid`	Mode specific	`-bottom_selid`	Mode specific
`-head_selid`	Mode specific	`-judgeup`	Mode specific
`-nmolup`	Mode specific	`-center_selid`	Mode specific
`-dx`	Mode specific	`-dt`	Mode specific
`-prep_only`	Common	–	–

- **-sel**x **<selection>** (up to 7 selections)
  - Default: N/A
  - Example: `-sel0 name P -sel1 name N`

  Specify the atomic groups used to define the bottom and head parts of the molecule. If using a reference molecular orientation to define the z-axis (`-zdef_type outerprod`), additional selections for $\mathbf{v}_0$ and $\mathbf{v}_1$ must also be provided. If you use `-judgeup`, selection of reference molecules to determine direction is also necessary.

- **-head_selid**, **-bottom_selid**
  - Default: N/A
  - Example: `-head_selid 0 -bottom_selid 1`

  Specify which selections correspond to the head and bottom of the molecule used to compute the orientation vector $\mathbf{u}$.

- **-zdef_type <coord or outerprod>**,
  - Default: N/A
  - Example: `-zdef_type coord`

  Required for specifying the definition of the z-direction.

  - `coord`: Use laboratory frame z-axis
  - `outerprod`: Use cross product of vectors $\mathbf{v}_0$ and $\mathbf{v}_1$ to define z-axis

  If `outerprod`, the following options smust be set.

  - `-vec0_bottom_selid` $\rightarrow \mathrm{x}_{\mathrm{B}_0}$

- `-vec0_head_selid` $\to \mathrm{x}_{\mathrm{H}_0}$
- `-vec1_bottom_selid` $\to \mathrm{x}_{\mathrm{B}_1}$
- `-vec1_head_selid` $\to \mathrm{x}_{\mathrm{H}_1}$

- **-judgeup <nmolup coord or none>**
  - Default: `coord`
  - Example: `-judgeup coord`
- **-nmolup <# of target molecules in upper leaflet>**
  - Default: 0
  - Example: `-nmolup 100`
- **-center_selid <selid>**
  - Default: N/A
  - Example: `-center_selid 2`

This option specifies the orientation of $\mathbf{e}_z$ when taking the laboratory coordinate system′s z-axis as the reference. The unit vector pointing in the positive z-axis direction is denoted as $\mathbf{e}'_z$.

- In the case of `-judgeup nmolup`: Let N_{up} be the number specified by the `-nmolup` option. Then, the first N_{up} molecules are assigned $\mathbf{e}_z = \mathbf{e}'_z$ for calculating $\cos\theta$, and the remaining molecules are assigned $\mathbf{e}_z = -\mathbf{e}'_z$ for calculating $\cos\theta$.
- In the case of `-judgeup coord`: Let z_{c} be the z-coordinate of the center of mass of the molecule specified by `-center_selid`, and z_{B} and z_{H} be the z-coordinates of the centers of mass of the regions specified by `-bottom_selid` and `-head_selid`, respectively. Then,

$$\mathbf{e}_z = \begin{cases} \mathbf{e}'_z & \dfrac{z_{\mathrm{H}} + z_{\mathrm{B}}}{2} > 0 \\ -\mathbf{e}'_z & \dfrac{z_{\mathrm{H}} + z_{\mathrm{B}}}{2} \leq 0 \end{cases} \tag{3.50}$$

- **-dx <grid width for $\cos\theta$>**
  - Default: `0.1`
  - Example: `-dx 0.1`

  Grid width $\Delta\chi$ used for computing $P(\cos\theta)$.

- **-dt <time step>**
  - Default: `1.0`

  - Example: `-dt 0.1`

Time increment for the output time series data.

3.12.2 Output File Format

- `*.costheta`
  Time series data of $\cos\theta$:
  - Column 1: Time
  - Column 2: $\cos\theta$ of molecule 1
  - Column 3: $\cos\theta$ of molecule 2
    …
- `*.distr`
  Probability distribution $P(\cos\theta)$:
  - Column 1: $\cos\theta$
  - Column 2: $P(\cos\theta)$

3.12.3 Examples of Analysis

3.12.3.1 Orientation Distribution of Lipids in a Bilayer

```
#!/usr/bin/env bash

anatra z_orient                    \
  -stype       psf                 \
  -sfile       complex.psf         \
  -tintype     nc                  \
  -flist_traj  flist               \
  -fhead       run_nc              \
  -judgeup     coord               \
  -dx          0.1                 \
  -dt          0.1                 \
  -sel0        name P              \
  -sel1        name N              \
  -sel2        segid MEMB and noh  \
  -mode0       residue             \
  -mode1       residue             \
  -mode2       whole               \
  -bottom_selid 0                  \
```

(continues on next page)

(continued from previous page)

```
  -head_selid   1                   \
  -center_selid 2                   \
  -prep_only   false
```