

# gqteaWinToga User Manual

## Index

- [Overview](#)
- [Installation and Requirements](#)
  - [Requirements](#)
    - [Python packages](#)
    - [System requirements that pip cannot install](#)
    - [Optional packages](#)
    - [External simulation programs](#)
  - [Install from Source](#)
  - [Start the Program](#)
- [Input File Basics](#)
  - [XYZ Files](#)
  - [Atom Labels](#)
  - [Common Units](#)
- [Basic Workflows](#)
  - [Analyze a Bond Length](#)
  - [Build CPMD Input Files](#)
  - [Select Frames and Generate Gaussian Inputs](#)
  - [Plot CPMD Energy Data](#)
- [Features and Commands](#)
  - [Main Window](#)
  - [Geometry Tools](#)
    - [Bond Length Analysis](#)
    - [All Bond Distance Analysis](#)
    - [Bond Angle Analysis](#)
    - [All Bond Angle Analysis](#)
    - [Dihedral Angle Analysis](#)
    - [All Dihedral Angle Analysis](#)
    - [Hydrogen Bond Analysis](#)
    - [Comparison of Molecular Geometric Parameters](#)
  - [Input Builders](#)
    - [CPMD Inputs](#)
    - [Surface Hopping Input Builder](#)
    - [Collision Input](#)
    - [Quantum ESPRESSO cp.x Input Builder](#)
    - [ORCA Input Builder](#)
    - [gqteaMD Input Builder](#)

- [Vanderbilt runatom.x Input Builder](#)
- [Structural Tools](#)
  - [Radial Distribution Function](#)
  - [Mean Residence Time](#)
  - [Legacy Mean Residence Time](#)
  - [Autocorrelation Function](#)
  - [Single Solute Solvent Box](#)
  - [Mixture of Two Solvent Box](#)
  - [Shared-Wall Double Solvent Box](#)
- [Thermo Tools](#)
  - [Classical Rate Constant](#)
- [General Tools](#)
  - [3D Molecular Viewer](#)
    - [Molecular Design](#)
      - [Opening and using the two windows](#)
      - [Creating and modifying bonds](#)
      - [Running the pre-optimization](#)
      - [Validation messages](#)
  - [Energy Plots](#)
  - [Molecular Axis Alignment](#)
  - [Select Frames](#)
  - [Frame Selection by Interatomic Distance Range](#)
  - [CPMD Input to XYZ Converter](#)
  - [SH Geometry Analyzer](#)
  - [Convert cp.x .pos File to trajec.xyz](#)
  - [Compute Forces from cp.x .for File](#)
  - [Coordinate Converter](#)
  - [Convert Coordinates from Angstrom to Bohr](#)
  - [Unit Converter](#)
- [Configuration](#)
- [Practical Examples](#)
  - [Example 1: Find a Bond-Length Free-Energy Minimum](#)
  - [Example 2: Prepare a CPMD Collision Restart](#)
  - [Example 3: Group Surface Hopping Frames by State](#)
  - [Example 4: Convert a cp.x Trajectory](#)
  - [Example 5: Analyze All Solute Bond Distances](#)
  - [Example 6: Analyze All Solute Bond Angles](#)
  - [Example 7: Analyze All Solute Dihedral Angles](#)
- [Formulas and Methods \(All Bond / Angle / Dihedral tools\)](#)
- [Formulas and Methods \(Bond Length / Bond Angle / Dihedral Angle tools\)](#)
- [Formulas and Methods \(Mean Residence Time tool\)](#)

- [Troubleshooting](#)
  - ["No file was selected!"](#)
  - ["Please input a valid value for ..."](#)
  - ["Invalid format for ... Please input exactly two atom labels."](#)
  - ["Failed to open file" or "Failed to read file"](#)
  - ["The file is empty!"](#)
  - ["Invalid line format in TRAJECTORY file"](#)
  - ["The stop frame must be less than the total number of frames"](#)
  - [ENERGY file format errors](#)
  - [Solvent box insertion warnings](#)
  - [Missing `SH\_STATE.dat` or `TRAJEC.xyz`](#)
  - [Plot windows do not appear](#)
  - [Molecular viewer does not open or shows no 3D view](#)
  - [Packaged Windows executable cannot find GLFW](#)
- [Version, Contributors, and License](#)

## Overview

gqteaWinToga is a desktop molecular analysis toolkit developed by the gQTEA group. It helps users prepare, inspect, convert, visualize, and analyze molecular simulation data from CPMD, Quantum ESPRESSO [cp.x](#), ORCA, surface hopping workflows, and related molecular dynamics tools.

The program is a graphical application built with Python and Toga. Most tools follow the same pattern: open a tool window from the main tabbed interface, browse to one or more input files, enter calculation parameters, then click the tool's action button to generate outputs.

The current application title identifies the toolkit as:

```
gQTEA-0.6.0 Molecular Analysis Toolkit
```

## Installation and Requirements

### Requirements

Recommended environment:

- Python 3.10 or newer. The toolkit is developed and verified on Python 3.13.
- Windows with a working graphical desktop session.
- A Python virtual environment.

## Python packages

All Python dependencies are listed in [installation\\_requirements.txt](#), in the source directory. Install them in one step:

```
python -m pip install -r installation_requirements.txt
```

The file installs these packages:

| Package                             | Used for                                                                                                                                                     |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="#">toga-winforms</a>       | The GUI framework and its Windows backend. Also installs <a href="#">toga-core</a> , <a href="#">pythonnet</a> and <a href="#">pillow</a> .                  |
| <a href="#">numpy</a>               | Trajectory parsing, geometry and statistics, used by most analysis tools.                                                                                    |
| <a href="#">scipy</a>               | The autocorrelation FFT, the molecular pre-optimizer, and the solvent-box builders.                                                                          |
| <a href="#">matplotlib</a>          | All plotting, in both the main window and the separate interactive plot viewer.                                                                              |
| <a href="#">glfw</a>                | Windowing and mouse input for the 3D molecular viewer.                                                                                                       |
| <a href="#">PyOpenGL</a>            | OpenGL bindings for the 3D molecular viewer.                                                                                                                 |
| <a href="#">PyOpenGL-accelerate</a> | Optional speedups for <a href="#">PyOpenGL</a> . The toolkit runs correctly without it, only more slowly in the 3D viewer.                                   |
| <a href="#">PyMuPDF</a>             | Renders <a href="#">cpx_input_description.pdf</a> inside the <a href="#">cp.x</a> input builder's help viewer. Without it, only the in-app PDF help is lost. |

Verified working on Python 3.13.12 with [toga-winforms](#) 0.5.3, [numpy](#) 2.4.2, [scipy](#) 1.17.0, [matplotlib](#) 3.10.8, [glfw](#) 2.10.0, [PyOpenGL](#) 3.1.10, [PyOpenGL-accelerate](#) 3.1.10 and [PyMuPDF](#) 1.27.2.2.

Two native libraries that the 3D viewer needs — [glfw3.dll](#) and [freeglut](#)— are bundled inside the [glfw](#) and [PyOpenGL](#) wheels, so a normal pip install requires no separate download. A packaged [.exe](#) build is different; see [Packaged Windows executable cannot find GLFW](#).

## System requirements that pip cannot install

Three things are not Python packages and cannot be installed from [installation\\_requirements.txt](#). They are listed as comments at the end of that file as well.

## 1. tkinter (Tcl/Tk)

A standard-library module that is not distributed on PyPI. The interactive plot viewer uses matplotlib's [TkAgg](#) backend, which requires it.

On Windows, tkinter is included with the official python.org installer as long as the **tcl/tk and IDLE** component is selected. To add it to an existing installation, re-run the installer, choose **Modify**, and enable that component. Check whether it is present with:

```
python -c "import tkinter; print(tkinter.TkVersion)"
```

If tkinter is missing, plot windows fail to open **silently**, with no error message. See [Plot windows do not appear](#).

## 2. A graphical desktop session with working OpenGL drivers

The 3D molecular viewer opens a native OpenGL window. It cannot run over a plain SSH session, in a headless container, or on a machine with no GPU driver installed. Install the vendor driver for your graphics hardware (Intel, NVIDIA or AMD); the Microsoft Basic Display Adapter driver that Windows falls back to does not provide an OpenGL runtime. Remote Desktop sessions may work through a software renderer, but slowly.

## 3. Microsoft .NET runtime

[toga-winforms](#) builds the interface on WinForms through [pythonnet](#). Windows 10 and 11 ship a suitable .NET Framework, so no action is normally needed. A stripped-down or Server Core installation may not include it.

## Optional packages

Neither is needed to run the toolkit.

To run the automated test suite:

```
python -m pip install pytest
python -m pytest tests/ -q
```

To build the standalone Windows executable:

```
python -m pip install auto-py-to-exe
```

Both are listed as commented entries in [installation\\_requirements.txt](#).

## External simulation programs

Some workflows also require files produced by external chemistry programs, such as CPMD, Quantum ESPRESSO [cp.x](#), ORCA, Gaussian, or Vanderbilt [runatom.x](#). gqteaWinToga prepares and analyzes files for these programs, but it does not replace the external simulation engines.

## Install from Source

From PowerShell:

```
git clone https://github.com/AdemirCamargo57/gqteaWinToga.git
cd gqteaWinToga
python -m venv venv
.\venv\Scripts\Activate.ps1
python -m pip install --upgrade pip
python -m pip install -r installation_requirements.txt
```

If you already have the source tree and an active virtual environment, run the last command in that environment.

To confirm the installation before starting the program:

```
python -c "import toga, numpy, scipy, matplotlib, glfw, OpenGL, fitz; print('all packages import correctly')"
python -c "import tkinter; print('tkinter', tkinter.TkVersion)"
```

## Start the Program

Run the launcher from the source directory:

```
python gqteaWinToga.py
```

The main window opens with five tabs:

- **Geometry**
- **Inputs**
- **Structural**
- **Thermo**
- **Tools**

Choose a tab, click a tool, complete the fields in the tool window, and run the calculation or conversion.

## Input File Basics

### XYZ Files

Many tools read XYZ geometry or trajectory files. A standard single-frame XYZ file has:

```
5
Comment line
P  0.112  -0.092   0.064
O  0.112   1.475  -0.490
O  1.470  -0.876  -0.490
O -1.246  -0.876  -0.490
O  0.112  -0.092   1.727
```

The first line is the number of atoms. The second line is a comment. Each remaining line contains an atomic symbol and Cartesian coordinates.

Trajectory XYZ files repeat this block for each frame. CPMD trajectory analysis tools commonly expect a file named [TRAJEC.xyz](#).

## Atom Labels

Most analysis tools use 1-based atom labels. For example, 1 2 means atom 1 and atom 2 in the XYZ frame, not Python-style zero-based indices.

## Common Units

- Distances: Angstrom.
- Angles: degrees.
- Temperature: kelvin.
- CPMD time step: atomic units unless a tool states otherwise.
- Some collision outputs distinguish between CPMD atomic-unit velocities and gqteaMD Angstrom/fs velocities.

## Basic Workflows

### Analyze a Bond Length

- 1 Start gqteaWinToga.
- 2 Open **Geometry**.
- 3 Click **Bond length analysis**.
- 4 Browse to TRAJEC.xyz.
- 5 Enter the maximum distance for the distribution, time step, sampling interval, temperature, two atom labels, and histogram bin width.
- 6 If the trajectory is periodic and unwrapped, enter **Cell lengths a b c**; leave it blank for an isolated system.
- 7 Keep **Show plots at the end** enabled if you want plot windows.
- 8 Keep **Save CSV outputs** enabled if you want CSV files in addition to .dat files.
- 9 Enable **Jacobian r<sup>2</sup> correction (PMF)** if you want a potential of mean force rather than -RT ln P.
- 10 Click **Analyze**.

Typical outputs include bond length time series, distribution data, free-energy data, summary text, and optional CSV files. Progress appears as text in the output box, and the run ends by showing a summary of the parameters used and the results obtained. Figures open in the interactive viewer; no PNG files are written.

## Build CPMD Input Files

- 1 Open **Inputs**.

- 2 Click **CPMD Inputs**.
- 3 Enter a prefix, total charge, periodic cell parameters, energy cutoff, and dual value.
- 4 Select a cell symmetry.
- 5 Browse to an XYZ starting geometry.
- 6 Click **Input Builder**.

The generated CPMD input files are based on the selected geometry and values entered in the form.

## Select Frames and Generate Gaussian Inputs

- 1 Open **Tools**.
- 2 Click **Select Frames**.
- 3 Enter the starting frame, number of skipped frames, and stop frame.
- 4 Browse to [TRAJEC.xyz](#).
- 5 Optionally enable Gaussian input generation and fill in checkpoint name, memory, processor count, route section, charge, and multiplicity.
- 6 Click **Select Frames**.

The tool writes [selected\\_frames.xyz](#). If Gaussian generation is enabled, it also writes files named like [gl6 input 1.gjf](#), [gl6 input 2.gjf](#), and so on.

## Plot CPMD Energy Data

- 1 Open **Tools**.
- 2 Click **Energy plots**.
- 3 Select the energy-file type.
- 4 Browse to the energy file.
- 5 Enter the simulation time step if needed.
- 6 Select the x-axis unit and one or more plot types.
- 7 Click **Plot**.

For CPMD ENERGY files, the program expects a valid numeric table with eight columns.

## Features and Commands

The program is GUI-based. The "commands" below refer to the buttons and controls in each tool window.

## Main Window

The main window organizes tools by category:

- **Geometry:** bond length, all bond distances, bond angle, all bond angles, dihedral angle, all dihedral angles, and hydrogen bond analysis.
- **Inputs:** CPMD, surface hopping, collision, Quantum ESPRESSO [cp.x](#), ORCA, gqteaMD, and Vanderbilt [runatom.x](#) input builders.
- **Structural:** radial distribution function, mean residence time, autocorrelation function, and solvent box builders.
- **Thermo:** classical rate constant calculation.
- **Tools:** visualization, plotting, frame selection, converters, force processing, coordinate tools, and unit conversion.

## Geometry Tools

### Bond Length Analysis

Use this tool to calculate bond lengths across a trajectory, average bond length, bond-length distribution, and Helmholtz free energy from the probability distribution.

Inputs:

- Maximum [r](#) for the distribution function.
- Simulation time step in atomic units.
- Sampling interval in frames.
- Simulation temperature in K.
- Two atom labels, for example [1 2](#).
- Histogram bin width in Angstrom.
- Optional **Cell lengths a b c** in Angstrom.
- [TRAJEC.xyz](#) trajectory file.

Options:

- **Show plots at the end:** opens plot windows after the run.
- **Save CSV outputs:** writes CSV versions of the main data tables.
- **Jacobian  $r^2$  correction (PMF):** see below.

Periodic boundaries:

- If **Cell lengths a b c** is filled, the distance uses the orthorhombic **minimum-image convention**; leave it blank for an isolated or already-wrapped system.
- Minimum image is exact only up to half the smallest box length, so with cell lengths set, the maximum  $r$  must not exceed  $\min(a, b, c) / 2$ . The tool refuses to run otherwise.
- Coordinates do not need to be wrapped into the box.

Free energy:

- By default the tool reports  $G(r) = -RT \ln P(r)$ , where  $P(r)$  is the fraction of frames falling in each bin.
- With **Jacobian  $r^2$  correction (PMF)** enabled it reports the potential of mean force  $W(r) = -RT \ln[P(r)/r^2]$ , which removes the  $4\pi r^2$  volume-element bias of sampling a separation in three dimensions. Both are defined only up to an additive constant, so compare shapes and well depths, not absolute values.

Bond lengths at or beyond the maximum  $r$  are excluded from the distribution and from the free energy. When that happens the tool reports how many frames were dropped, and records the count in the summary file.

Progress reporting:

- There is no progress bar and no frame-count label. Loading the trajectory and running the calculation both report as text in the large output box.
- When the run finishes, the box shows only a summary: a **PARAMETERS** block listing the values used, and a **RESULTS** block with the bond statistics, the lowest free energy, any excluded bonds, and the output directory. The same text is saved as  $\text{summary } \langle \text{atom1} \rangle \langle \text{atom2} \rangle .\text{txt}$ .

Typical outputs:

- $\text{bond } \langle \text{atom1} \rangle \langle \text{atom2} \rangle .\text{dat}$
- $\text{bond distribution } \langle \text{atom1} \rangle \langle \text{atom2} \rangle .\text{dat}$
- $\text{free energy } \langle \text{atom1} \rangle \langle \text{atom2} \rangle .\text{dat}$
- $\text{summary } \langle \text{atom1} \rangle \langle \text{atom2} \rangle .\text{txt}$
- Optional CSV files with matching names.

Plots open in the interactive viewer (zoom, pan, and save from its toolbar). No static PNG image files are written next to the trajectory; use the viewer's save button if you want an image.

## All Bond Distance Analysis

Use **Geometry > All bond distance analysis** to calculate distance statistics for every connected atom pair in an XYZ trajectory. This tool is intended for complete solute geometry monitoring in a molecular dynamics run. For a solute in a solvent box, provide the solute atom indices so solvent-solvent and solute-solvent distances are ignored.

How connectivity is detected:

- Connectivity is re-evaluated **on every frame**: in each frame, two in-scope atoms are bonded when their distance is less than or equal to the **Maximum connection distance**.
- The reported pair set is the **union** of every pair bonded in at least one frame, so bonds that form or break mid-run are captured (not only those present in the first frame).
- Each pair's average distance, variance, and standard deviation are computed **only over the frames in which it is bonded**, and the tool also reports the **occurrence** (fraction of frames the pair is bonded).
- The default maximum connection distance is 1.7 Angstrom.
- If **Solute atom indices** is filled, only pairs where both atoms belong to the solute are considered.
- If **Cell lattices a b c** is filled, distances use the orthorhombic **minimum-image convention** (periodic boundaries); leave it blank for an isolated system.

Inputs:

- trajectory.xyz or TRAJEC.xyz trajectory file.
- **Maximum connection distance (A)** — leave blank to use the default 1.7.
- **Cell lattices a b c (A)** — three positive box lengths for PBC; leave blank for no periodic boundaries. When used, the connection distance must not exceed half the smallest lattice.
- **Solute atom indices** — 1-based labels; **range syntax is supported**, e.g. 1-5 14-16 18 20 is equivalent to 1 2 3 4 5 14 15 16 18 20. Whitespace and commas both separate tokens. Leave blank to use all atoms.
- Output TXT filename, default all bond analysis combined.txt.

Step-by-step:

- 1 Open **Geometry > All bond distance analysis**.
- 2 Click **Browse** and select the trajectory XYZ file.
- 3 Enter the maximum connection distance, or leave it blank for the 1.7 default.
- 4 To apply periodic boundaries, enter the cell lattices a b c; otherwise leave blank.
- 5 If the system contains a solute in solvent, enter the solute atom labels in **Solute atom indices** (ranges allowed, e.g. 1-5 14-16 18 20).

- 6 Enter the output filename or keep all\_bond\_analysis\_combined.txt.
- 7 Click **Analyze**.
- 8 Wait until the final summary appears in the text area.
- 9 Open the statistics file and pair mapping file in the trajectory directory.

Output file:

The tool writes a **single self-describing file**, all\_bond\_analysis\_combined.txt, that merges the pair identity, distance statistics, and run metadata. It begins with a commented header block:

```
# gQTEA All Bond Distance Analysis
# frames_used <N>
# max_connection_distance <cutoff>
# atom_scope all_atoms (or: # atom_scope solute_atoms + #
solute_atom_indices ...)
# periodic_boundary none (or: # cell_lengths a b c)
# row atom_i atom_j element_i element_j first_bonded_distance
average variance standard_deviation occurrence_fraction
frames_bonded
```

Each following row is one connected atom pair (atom labels 1-based):

```
row atom_i atom_j element_i element_j first_bonded_distance
average variance standard_deviation occurrence_fraction
frames_bonded
```

Because the data rows include the two element-symbol columns, load the numeric columns with `np.loadtxt(path, usecols=(0,1,2,5,6,7,8,9,10))`, or use `np.genfromtxt(path, dtype=None, names=True, comments='#')` to keep the element strings.

## Bond Angle Analysis

Use this tool to calculate bond angles across a trajectory, angle distributions, and free-energy profiles.

Inputs:

- Maximum angle for the distribution function.
- Simulation time step in atomic units.
- Sampling interval in frames.
- Simulation temperature in K.
- Three atom labels i j k, for example 1 2 3. The vertex is the middle label. All three must be different.
- Histogram bin width in degrees.
- Optional **Cell lengths a b c** in Angstrom.
- TRAJEC.xyz trajectory file.

Options:

- **Show plots at the end.**
- **Save CSV outputs.**
- **Use sin(theta) Jacobian:** reports the potential of mean force  $W(\theta) = -RT \ln[P(\theta)/\sin(\theta)]$  instead of  $-RT \ln P(\theta)$ , removing the volume-element bias of sampling an angle in three dimensions. Off by default.
- **Smooth free energy:** applies a small moving average to the probabilities before the logarithm, which tames noise at fine bin widths. **On by default** for this tool. Turn it off for the raw curve.

Periodic boundaries:

- If **Cell lengths a b c** is filled, both arms of the angle use the orthorhombic **minimum-image convention**; leave it blank for an isolated or already-wrapped system.
- Minimum image is exact only for arms shorter than  $\min(a, b, c)/2$ . The tool warns when an arm exceeds that rather than failing silently.

Angles at or beyond the maximum angle are **excluded and reported**, not clamped into the last bin. The run ends showing only a summary: a **PARAMETERS** block and a **RESULTS** block, also saved as summary <i> <j> <k>.txt. Progress appears as text in the output box; there is no progress bar or frame-count label. Figures open in the interactive viewer, and no PNG files are written.

## All Bond Angle Analysis

Use **Geometry > All bond angle analysis** to calculate angle statistics for every connected atom i-atom j-atom k angle in an XYZ trajectory. The central atom is atom j. Connectivity is **re-evaluated on every frame**: a triplet exists in a frame when both of its bonds (atom j-atom i and atom j-atom k) are within the cutoff. The reported set is the union of every triplet that appears in at least one frame, each triplet's statistics are averaged only over the frames where it exists, and the tool reports the **occurrence** (fraction of frames the triplet is connected).

For a solute in a solvent box, provide the solute atom indices. The program then builds angles only from solute atoms, excluding any angle involving solvent atoms.

Inputs:

- [trajectory.xyz](#) or [TRAJEC.xyz](#) trajectory file.
- **Maximum connection distance (A)** — leave blank to use the default [1.7](#).
- **Cell lattices a b c (A)** — three positive box lengths for PBC (orthorhombic minimum-image); leave blank for no periodic boundaries.
- **Solute atom indices** — 1-based labels; **range syntax is supported**, e.g. [1-5](#) [14-16](#) [18](#) [20](#). Leave blank to use all atoms.
- Output TXT filename, default [all angle analysis combined.txt](#).

Step-by-step:

- 1 Open **Geometry > All bond angle analysis**.
- 2 Click **Browse** and select the trajectory XYZ file.
- 3 Enter the maximum connection distance, or leave it blank for the [1.7](#) default.
- 4 To apply periodic boundaries, enter the cell lattices [a b c](#); otherwise leave blank.
- 5 Enter solute atom labels if solvent atoms must be excluded (ranges allowed).
- 6 Enter the output filename or keep [all angle analysis combined.txt](#).
- 7 Click **Analyze**.
- 8 Wait until the text area reports that the analysis is completed.
- 9 Inspect the combined output file.

Output file:

A single self-describing file, [all angle analysis combined.txt](#), with a commented metadata header ([# frames used](#), [# max connection distance](#), [# atom scope](#), and [# periodic boundary none](#) / [# cell lengths a b c](#)) followed by one row per triplet:

```
# row atom_i atom_j atom_k element_i element_j element_k
first_present_angle_degrees average_angle_degrees variance_degrees2
standard_deviation_degrees occurrence_fraction frames_present
```

Atom labels are 1-based. Because the rows include element-symbol columns, load numeric columns with `np.loadtxt(path, usecols=...)` or `np.genfromtxt(path, dtype=None, names=True, comments='#')`.

## Dihedral Angle Analysis

Use this tool to calculate dihedral angles across a trajectory, dihedral distributions, and free-energy profiles.

Inputs:

- Maximum angle for the distribution.
- Simulation time step in atomic units.
- Sampling interval in frames.
- Simulation temperature in K.
- Four atom labels `i j k l`, for example `1 2 3 4`. All four must be different.
- Histogram bin width in degrees.
- Optional **Cell lengths a b c** in Angstrom.
- `TRAJEC.xyz` trajectory file.

Options:

- **Show plots at the end.**
- **Save CSV outputs.**
- **Wrap dihedral to [-180, 180]:** reports signed dihedral angles instead of a 0 to 360 degree range. With it on, the histogram spans `[-180, 180]` so negative angles are binned properly; with it off, angles map to `[0, 360)` and the histogram spans `[0, maximum angle]`.
- **Smooth free energy:** applies a small moving average to the probabilities before the logarithm. **On by default** for this tool.

There is deliberately **no Jacobian option** here. Unlike a bond length (`4*pi*r^2`) or a bond angle (`sin(theta)`), a dihedral has a uniform volume element, so there is nothing to divide out: `-RT ln P(phi)` is already the potential of mean force. The summary records this as `Jacobian: not applicable (uniform measure)`.

Periodic boundaries:

- If **Cell lengths a b c** is filled, all three connecting vectors use the orthorhombic **minimum-image convention**; leave it blank for an isolated or already-wrapped system.

- Minimum image is exact only for bonds shorter than  $\min(a,b,c)/2$ . The tool warns when one exceeds that.

Angles outside the histogram range are **excluded and reported**, not clamped. The run ends showing only a summary, also saved as `summary <i> <j> <k> <l>.txt`. Progress appears as text in the output box; there is no progress bar or frame-count label. Figures open in the interactive viewer, and no PNG files are written.

## All Dihedral Angle Analysis

Use **Geometry > All dihedral angle analysis** to calculate statistics for every connected `atom i-atom j-atom k-atom l` dihedral path in an XYZ trajectory. Connectivity is **re-evaluated on every frame**: a quadruplet exists in a frame when all three of its bonds (`atom i-atom j`, `atom j-atom k`, `atom k-atom l`) are within the cutoff. The reported set is the union of every quadruplet that appears in at least one frame, each quadruplet's statistics are averaged only over the frames where it exists, and the tool reports the **occurrence** (fraction of frames the quadruplet is connected).

For a solute in a solvent box, provide the solute atom indices. The program then builds only solute dihedrals and ignores all solvent atoms.

Dihedral convention:

- Dihedrals are signed angles in degrees.
- The range is `[-180, 180]`.
- This convention matches the single **Dihedral angle analysis** tool.

Inputs:

- `trajectory.xyz` or `TRAJEC.xyz` trajectory file.
- **Maximum connection distance (A)** — leave blank to use the default **1.7**.
- **Cell lattices a b c (A)** — three positive box lengths for PBC (orthorhombic minimum-image); leave blank for no periodic boundaries.
- **Solute atom indices** — 1-based labels; **range syntax is supported**, e.g. `1-5 14-16 18 20`. Leave blank to use all atoms.
- Output TXT filename, default `all_dihedral_analysis_combined.txt`.

Step-by-step:

- 1 Open **Geometry > All dihedral angle analysis**.
- 2 Click **Browse** and select the trajectory XYZ file.

- 3 Enter the maximum connection distance, or leave it blank for the 1.7 default.
- 4 To apply periodic boundaries, enter the cell lattices a b c; otherwise leave blank.
- 5 Enter solute atom labels if solvent atoms must be excluded (ranges allowed).
- 6 Enter the output filename or keep all\_dihedral\_analysis\_combined.txt.
- 7 Click **Analyze**.
- 8 Wait until the text area reports that the analysis is completed.
- 9 Inspect the combined output file.

Output file:

A single self-describing file, all\_dihedral\_analysis\_combined.txt, with a commented metadata header (# frames used, # max connection distance, # dihedral convention signed degrees minus180 to 180, # atom scope, and # periodic boundary none / # cell lengths a b c) followed by one row per quadruplet:

```
# row atom_i atom_j atom_k atom_l element_i element_j element_k
element_l first_present_dihedral_degrees average_dihedral_degrees
variance_degrees2 standard_deviation_degrees occurrence_fraction
frames_present
```

Atom labels are 1-based. Because the rows include element-symbol columns, load numeric columns with np.loadtxt(path, usecols=...) or np.genfromtxt(path, dtype=None, names=True, comments='#').

## Hydrogen Bond Analysis

Use this tool to analyze hydrogen bonds from a trajectory.

Inputs:

- Maximum hydrogen bond length.
- Simulation time step in atomic units.
- Sampling interval in frames.
- Simulation temperature in K.
- Histogram bin widths for hydrogen bond length and angle.
- Donor atom label, hydrogen atom label, and acceptor atom label.
- TRAJEC.xyz trajectory file.

Click **Exec** or the main action button in the tool window to run the calculation.

## Comparison of Molecular Geometric Parameters

Use this tool to see how a different environment changes a molecule's geometry. It compares the geometric parameters of the **same molecule** taken from two simulations — typically the isolated molecule and the same molecule in a water box, with the solvent already removed from the trajectory.

Inputs:

- **Parameter file 1 (reference)** and **Parameter file 2 (comparison)**: two combined parameter files written by the All bond distance, All bond angle or All dihedral angle analysis tools. Both must be of the same type; the type is read from each file's own title line.
- **Label for file 1 / Label for file 2** (default isolated and solvated): these become part of the output column names, e.g. average isolated.
- **Minimum occurrence fraction** (blank = 0, keep everything): ignores parameters present in fewer than this fraction of frames.
- **Relative difference columns** — two independent switches, either, both, or neither on:
  - **Symmetric percent difference** (on by default):  $100 * (\text{avg } 2 - \text{avg } 1) / \text{mean}(\text{avg } 1, \text{avg } 2)$ .
  - **Percent error vs file 1** (off by default):  $100 * (\text{avg } 2 - \text{avg } 1) / \text{avg } 1$ .
- Both are signed, file 2 minus file 1. Leaving both off writes the output file exactly as it was before these columns existed.
- **Plot** — **Open the comparison figure after comparing** (on by default) and **Parameters to plot** (blank = 25, the parameters that shifted most).
- **Output folder** (blank = next to parameter file 1) and **Output txt filename** (blank = geometry comparison <type>.txt).

Each file is read as soon as you select it, so a wrong or corrupt file is reported immediately, together with its type, frame count, atom scope and molecule size.

Click **Compare** to run, **Help** for the full in-program description.

### *How atoms are matched*

Atoms are **not** matched by their raw index. A solvated run is normally analysed with solute atom indices, so its rows carry the atoms' indices in the **full box**: the oxygens of the reference molecule may be atoms 295, 296 and 297 there, while the same atoms are 29, 30 and 31 in the isolated run. The tool reads the # solute atom indices header and rennumbers

every atom back onto the molecule, so C1-O295 in the box is recognised as the same bond as C1-O29 isolated. The renumbering it applied is recorded in the output header.

Before comparing, the two files must agree on the parameter type, the molecule size, and the element of every atom. If any check fails the run is refused and the offending atom is named — a wrong atom mapping would produce results that look reasonable but are meaningless.

Parameters match regardless of atom order within a row: a bond is undirected, an angle keeps its vertex but may have its arms swapped, and a dihedral read backwards has the same signed value.

### *Why the occurrence filter matters*

The all-\* tools list every pair that came within the connection distance in at least one frame, so a long trajectory collects contacts that are not real bonds — typically H–H pairs sitting just under the 1.7 Å cutoff in a fraction of a percent of the frames. Setting the fraction to e.g. 0.05 drops them. The filter is applied to **both** files before matching, so a rare contact cannot be reported as a parameter that one simulation is missing.

### *Output*

A single text file with a metadata header (both source paths, labels, frame counts, atom scopes, the atom renumbering applied, the occurrence threshold and what it dropped, and the matched/unmatched counts) followed by three labelled sections:

```
# section matched
# row atom_i atom_j element_i element_j average_isolated
std_dev_isolated occurrence_isolated average_solvated
std_dev_solvated occurrence_solvated
difference_solvated_minus_isolated source_row_1 source_row_2
# section only_in_file_1
# row atom_i atom_j element_i element_j average std_dev occurrence
source_row
# section only_in_file_2
```

Angle and dihedral files carry three and four atom/element columns instead of two. Atom labels are the **molecule-local** 1-based indices; source row 1 / source row 2 give the row number the values came from in each input file, so any row can be traced back. Distances are in ångströms and angles in degrees. Dihedral differences are wrapped into (−180, 180], so a shift from +179° to −179° is reported as +2°, not −358°.

Every comment line starts with `#`, so a section can be read back with `np.genfromtxt(path, dtype=None, names=True, comments='#')` once it is separated from the others.

After a successful run the text box reports the counts, the output path, and the ten parameters whose average changed most between the two simulations.

### *Relative differences and global metrics*

With one or both **Relative difference columns** switches on, a percent column is appended after `difference solvated minus isolated` for each switch that is on — appended, never inserted earlier, so a reader that indexes the columns above by position keeps working:

```
# row ... difference_solvated_minus_isolated percent_difference
percent_error_vs_isolated source_row_1 source_row_2
```

`percent difference` is the symmetric form; `percent error vs isolated` carries the reference file's label so two identically labelled comparisons stay distinguishable. Both are signed, file 2 minus file 1. A percentage whose denominator is too close to zero to mean anything — below `1e-6 Å` for a bond, below `1.0°` for an angle or a dihedral — is written `nan`; for a dihedral the symmetric denominator is the circular mean of the two angles, not their arithmetic mean, so a shift between e.g.  $+179^\circ$  and  $-179^\circ$  does not divide by (almost) zero.

The header also carries the formula actually used, the denominator floor, the count of undefined rows, and global metrics over the whole matched set:

```
# formula_percent_difference 100*(avg_2-avg_1)/((avg_1+avg_2)/2)
# formula_percent_error 100*(avg_2-avg_1)/avg_1
# percent_denominator_floor 1e-06 angstrom
# percent_undefined_rows 0
# global_metric_category bond distance
# global_metric_n 42
# mae_angstrom 0.01234567
# rmsd_angstrom 0.01567890
# mae_percent_difference 1.23456789
# rmsd_percent_difference 1.56789012
```

`mae <unit>` / `rmsd <unit>` are the mean absolute error and root-mean-square deviation over every matched parameter, in its own unit (ångströms or degrees); the `mae percent *`

`/rmsd percent *` pair is added per selected switch, over the absolute value of that percentage, skipping undefined rows. With both switches off, none of the `formula *`, `percent *`, `global metric *`, `mae *` or `rmsd *` lines are written and the two extra columns are absent — the file is exactly what it always was.

### *What the message panel reports*

After a successful comparison the panel summarises both files, the molecule size, the counts of matched and unmatched parameters, and the global metrics, then lists the parameters that shifted most. Each switch that is on adds its percentage to every row of that list:

```
Global metrics over 6 matched bond distances:
MAE  0.01008 A      RMSD  0.01454 A
MAE  0.8198 %      RMSD  1.1746 %      (percent_difference)
MAE  0.8267 %      RMSD  1.1861 %      (percent_error)

Largest shifts (solvated - isolated):
C2-O3          1.41220 -> 1.44090      +0.02870 A
+2.01 %      +2.03 %
O3-H4          0.97120 -> 0.99060      +0.01940 A
+1.98 %      +2.00 %
C1-C2          1.52830 -> 1.53610      +0.00780 A
+0.51 %      +0.51 %
```

The percent columns appear in the same order as in the output file — symmetric difference first, then percent error — and a parameter whose percentage is undefined shows `nan` there without disturbing the rest of the row. With both switches off the list carries no percent columns at all.

### *The comparison figure*

With **Open the comparison figure after comparing** on, a grouped bar chart shows both files' averages for the **Parameters to plot** parameters that shifted most (ranked by the size of the shift), each bar carrying a  $\pm 1$  standard deviation error bar taken from its source file. If a percentage switch is on, that percentage is drawn as a line on a right-hand axis; with both switches on, the line always draws the symmetric percent difference, and the legend says so. The figure opens in the same interactive viewer (zoom/pan/save) as every other tool's plots, and no `.png` file is left next to the parameter files.

### *Reopening the figure later*

The viewer writes a small `.json` figure manifest into the output folder when it opens. To see the chart again without re-running the comparison, open **Tools > Plots**, set **Plot type** to `JSON plot file`, and select that file — the Plotter understands the bar-chart manifest and reopens the figure exactly as it was, error bars, percentage line and all.

## Input Builders

### CPMD Inputs

Use **Inputs > CPMD Inputs** to create CPMD input files from an XYZ geometry.

Fields:

- **Prefix file name:** prefix used for generated files, for example VitC.
- **Total charge on the system.**
- **Periodic box parameters:** a b c cosA cosB cosC, separated by spaces.
- **Energy Cutoff (Ry).**
- **Dual for rho expansion.**
- **Cell symmetry:**
  - 1 - CUBIC a=b=c alpha=beta=gamma=90
  - 6 - TETRAGONAL a=b!=c alpha=beta=gamma=90
  - 8 - ORTHORHOMBIC a!=b!=c alpha=beta=gamma=90
- XYZ input file.

Action:

- **Input Builder:** generates the CPMD input files.

### Surface Hopping Input Builder

Use **Inputs > SH Input Builder** to prepare CPMD surface hopping and TDDFT input files.

Fields:

- Starting frame.
- Number of skipped frames.
- Final frame.
- Number of electronic states.
- Initial state for SHTDDFT.
- Prefix file name.
- Molecular dynamics time step.
- Total system charge.
- Periodic box parameters.
- Energy cutoff.

- Dual for rho expansion.
- Cell symmetry.
- Maximum number of steps.
- [GEOMETRY.xyz](#) input file.
- [TRAJECTORY](#) input file.

Actions:

- **Read SH parms:** reads and displays the entered parameters.
- **SH Input Builder:** generates surface hopping input files.
- **Help:** opens the tool help text.

## Collision Input

Use **Inputs > Collision Input** to create new geometry/velocity files for collision molecular dynamics.

Fields:

- **MD engine:** select [cpmd](#) or [gqteaMD](#).
- **Attacker atom indices:** one or more 1-based atom labels, for example [12 13 14](#).
- **Initial velocity:** atomic units for CPMD; the label changes for gqteaMD.
- **Target xyz coordinates:** target point in Angstrom, for example [10.0 12.5 8.0](#).
- [GEOMETRY](#) or [GEOMETRY.xyz](#) file.

Options:

- **Zero velocities of non-attacker atoms.**
- **Compute center-of-mass KE.**

Typical outputs:

- [summary.txt](#)
- [newGeometry.xyz](#)
- [with-vibration-GEOMETRY](#)
- [without-vibration-GEOMETRY](#)
- [gqteaMD-GEOMETRY](#)
- [gqteaMD-without-vibration-GEOMETRY](#)

For CPMD collision simulations, the help text recommends restarting CPMD with:

RESTART WAVEFUNCTION COORDINATES VELOCITIES GEOFILE LATEST

## Quantum ESPRESSO cp.x Input Builder

Use **Inputs > cp.x Input Builder** to build Quantum ESPRESSO [cp.x](#) input files.

Main actions:

- **Open XYZ:** loads atomic coordinates from an XYZ file.
- **Preview input:** opens a preview of the generated input.
- **Save cp.x input:** writes the input file.
- **Help:** opens the bundled [cp.x input description.pdf](#).

Important behavior:

- If the output name is blank, the default name is based on the XYZ file, such as [<xyzname> cp.in](#).
- [ATOMIC SPECIES](#) is inferred from XYZ symbols.
- Default pseudopotential names use the pattern [<Element>.UPF](#); edit them if your pseudopotential filenames differ.
- [ATOMIC POSITIONS](#) can be written in [alat](#), [bohr](#), [crystal](#), or [angstrom](#).
- Optional sections include velocities, cell parameters, constraints, occupations, atomic forces, Wannier plotting, and autopilot text.

## ORCA Input Builder

Use **Inputs > ORCA Input Builder** to prepare ORCA [.inp](#) files from XYZ coordinates.

Main actions:

- **Open XYZ.**
- **Preview ORCA input.**
- **Save ORCA input.**
- **Clear form.**
- **Research notes.**

Common settings:

- Output filename. If blank, the default is <xyzname> orca.inp.
- Title/comment line.
- Method, basis, task, charge, multiplicity, and coordinate mode.
- Extra simple keywords.
- Parallel settings such as %pal nprocs.
- Memory setting such as %maxcore.
- SCF, geometry, CPCM, and advanced ORCA blocks.

## gqteaMD Input Builder

Use **Inputs > gqteaMD Input Builder** to generate a gqteaMD TOML input file.

Main actions:

- **Open XYZ.**
- **Preview TOML.**
- **Save TOML.**
- **Clear form.**
- **Manual notes.**

Common settings:

- Output filename. If blank, the default is <xyzname> gqteaMD.toml.
- Starting XYZ path.
- Force provider.
- Time step, number of steps, temperature, output intervals, restart behavior, and force-field sections.
- For xTB force provider, **OMP threads** writes omp num threads to the TOML so gqteaMD can set OMP NUM THREADS before xTB runs.
- Optional UFF or classical parameter blocks.

## Vanderbilt runatom.x Input Builder

Use **Inputs > Vanderbilt runatom.x Input Builder** to generate Vanderbilt runatom.x input files.

Controls:

- Mode selection for all-electron or generation input.
- **Also generate Makefile** option.
- **Generate** button.
- **Help** button.

Depending on the selected mode, the form shows the relevant atomic, pseudopotential, and state fields. The generated input is saved through a Save dialog, and a Makefile can be written alongside it.

## Structural Tools

### Radial Distribution Function

Use **Structural > Radial Distribution Function** to calculate the radial distribution function  $g(r)$  and the running coordination number around a chosen central atom, from a [TRAJEC.xyz](#) trajectory.

Inputs include:

- **Maximum radius for RDF** — the cutoff distance. It must not exceed half the smallest cell lattice; larger values are rejected (a periodic-boundary requirement). **Leave it blank to use the maximum allowed value automatically** (half the smallest cell lattice).
- **Bin width for histogram** — width of the distance bins (defaults to [0.01](#)).
- **Atom labels to be excluded** — space-separated 1-based labels to leave out, or [0](#) for none.
- **Shell center atom label** — the 1-based label of the central atom that  $g(r)$  is measured from.
- **Atomic symbol for  $g(r)$**  — the element counted as neighbours (e.g. [O](#)). It must be present in the trajectory.
- **Cell lattices (a b c)** — the three orthorhombic box lengths in Å.
- **X-axis range limit for RDF plot** — the x-max for the coordination-number plot. **Leave blank to default to half the a lattice** ([a / 2](#)).
- **Y-axis range limit for RDF plot** — the y-max for the coordination-number plot. **Leave blank to default to [10](#)**.
- **Select input file** — click **Browse** to load the [TRAJEC.xyz](#) trajectory (also counts the frames).

Coordinates do not need to be wrapped into the box; periodic images are handled internally by the minimum-image convention.

Actions:

- **Read Params:** reads and validates the settings and shows a summary. Invalid entries raise an explanatory dialog and stop.
- **RDF calculation:** runs the calculation. Live progress (loading, frame count) and a final completion summary — parameters used, cell volume and number density, the first  $g(r)$  peak, the coordination number, and the output file — are shown in the on-screen text area; malformed or truncated frames are reported.
- **Help:** shows usage instructions.
- **Close:** closes the window.

Progress is reported as text in the on-screen status area rather than a progress bar.

Typical outputs (written next to the trajectory):

- RDF <center><n> <symbol>.dat — three columns:  $r$  (bin center),  $g(r)$ , and the integrated coordination number.
- Interactive plots of  $g(r)$  and the coordination number.

## Mean Residence Time

Use **Structural > Mean Residence Time** for MRT analysis of an XYZ trajectory.

The trajectory field, the **Run MRT / Export Results / Clear Output** buttons, and the status line are always visible. The rest of the controls are grouped into four tabs.

### *Setup tab*

- Time step ( $\Delta t$  between saved frames) and its unit.
- Cutoff radius. Take it from the first minimum of the corresponding  $g(r)$  — the [Radial Distribution](#) tool will give you that.
- Tolerance frames ( $t^*$ ) — bridges short excursions: a particle that leaves for no more than  $t^*$  frames and returns is treated as never having left.
- **Cell lengths a b c** — optional orthorhombic periodic box. Leave blank for no periodic boundary; when given, distances use the minimum-image convention and the cutoff must not exceed half the shortest edge.
- **Scan  $t^*$  values** — optional list such as 0 1 2 5 10. The summary then includes one row per value so you can see how sensitive the result is before quoting a number.
- **Output folder** — optional; blank writes next to the trajectory.

Under **Advanced** on the same tab:

- **Integration rule** — how  $R(t)$  is turned into a time constant. *Truncate at first zero crossing* (default) integrates up to the point where the curve first goes negative; *Fit a single exponential* fits  $\ln R = a - t/\tau$  and reports  $\tau$ , which is what you want when  $R(t)$  has not decayed to zero inside the lag range; *Integrate the whole lag range* uses everything.
- **Maximum lag (frames)** — blank uses a tenth of the trajectory. Raise it if the correlation is still decaying at the end of the plotted range.
- **Censor events clipped by the start/end of the run** — on by default. Turn it off only to reproduce a number from an older version; leaving it off biases the mean residence time downwards.

The summary always states which rule was applied and how many lags it covered, so a run is reproducible from its own output. If it reports that all available lags were used, the curve never crossed zero and you should either raise the maximum lag or switch to the exponential fit.

### Definitions tab

- **Reference mode** — what the shell is centred on: a single atom, or the geometric centre or centre of mass of a group.
- **Observed mode** — what is watched: a single atom, every atom in a list tracked separately, or groups followed by their centre.

Each definition box carries an inline hint showing the exact syntax and a worked example for the mode currently selected, because the syntax changes with the mode (295 for one atom, 3-7, 10, 15-17 for a list, 1-3; 4-6 for groups). All atom indices are 1-based.

**Ranges.** Atom indices accept inclusive ranges as well as single values, so

3-7, 10, 15-17

means exactly the same as

3, 4, 5, 6, 7, 10, 15, 16, 17

Ranges and single indices can be mixed in any order, separated by commas or spaces, and they work inside group definitions too (1-3; 4-6). The order you type is preserved — useful because the tool labels observed objects in input order. This is the same syntax the [All Bond Distance Analysis](#) tool uses for its solute selection — one shared, tested parser.

Ranges must ascend and start at 1. Malformed input is rejected with a message naming the offending token:

| Input | Result                                                |
|-------|-------------------------------------------------------|
| 7-3   | Range bounds must be ascending: '7-3'.                |
| 0-5   | Atom indices must be positive integers starting at 1. |
| 3-    | Invalid index range: '3-'.                            |
| -5    | Invalid index range: '-5'.                            |
| 3-5-7 | Invalid index range: '3-5-7'.                         |
| a-b   | Invalid index range: 'a-b'.                           |

A bad token rejects the whole entry rather than being silently skipped, so 1, 2, 7-3, 9 is refused outright.

### Results tab

The full summary appears here after a run — parameters, results, the t\* scan table if you requested one, and notes on the definitions. **Show plots** opens three figures in the interactive viewer: the survival function S(t), the intermittent C(t) and R(t) overlaid, and the event-duration distribution. No image files are left next to your data.

### Help tab

The full reference for the tool, including what each reported quantity means.

### Output files

<name> continuous survival.dat,  
<name> intermittent correlation.dat,  
<name> mrt event durations.dat, and <name> mrt summary.txt.

### The two continuous time constants

The summary reports two different quantities, and they are not interchangeable:

- **Mean residence time** <T> — the average duration of a complete residence event, with a block-averaged uncertainty. Events that were already in progress when the trajectory started, or had not ended when it stopped, are *censored* (excluded) and counted separately, because including them at their truncated length biases <T> downwards, worst for the longest-lived species.
- **IMM survival integral** — the integral of the survival function averaged over time origins, in the sense of Impey, Madden and McDonald (*J. Phys. Chem.* **87** (1983) 5071). Mathematically this is the mean *residual* time; it equals <T> only for exponential kinetics and is larger for broad duration distributions.

The intermittent time constant from  $R(t)$  is integrated only over a bounded lag range (a tenth of the trajectory by default, truncated at the first zero crossing), because the longest lags average over very few time origins and are noise. The summary states which rule was applied and how many lags were used.

The tolerance time  $t^*$  has a strong effect on the result (Laage and Hynes, *J. Phys. Chem. B* **112** (2008) 7697), which is why the scan field exists — a residence time quoted from a single  $t^*$  should be treated with caution.

## Equations

Every equation the tool evaluates — with what it represents, where it is used, and its reference — is documented in [Formulas and Methods \(Mean Residence Time tool\)](#).

## A worked example

This example measures how long a water hydrogen stays hydrogen-bonded to a particular solute oxygen, in a periodic box of water.

**The system.** A [TRAJEC.xyz](#) from CPMD with 430 atoms per frame and 15121 frames: a C<sub>14</sub> solute in water, so the composition is C 14, H 280, O 136. The acceptor of interest is atom 295, an oxygen. The simulation cell is 15 × 15 × 22 Å.

**Choosing the cutoff.** Run **Structural > Radial Distribution Function** first, with atom 295 as the shell centre and **H** as the target. Its first peak is the hydrogen bond; the first minimum after that peak is where the first shell ends. For this system:

```
first O295-H peak : r = 1.905 Å
first minimum      : r = 2.405 Å    <- use this as the cutoff
```

## What to enter.

| Tab    | Field                   | Value                             | Why                                                              |
|--------|-------------------------|-----------------------------------|------------------------------------------------------------------|
| header | XYZ trajectory file     | <a href="#">C:\...\TRAJEC.xyz</a> |                                                                  |
| Setup  | Δt between saved frames | <a href="#">1.0</a>               | set your real value; <a href="#">1.0</a> reports times in frames |
| Setup  | unit                    | <a href="#">fs</a>                |                                                                  |
| Setup  | Cutoff radius (Å)       | <a href="#">2.405</a>             | the first g(r) minimum above                                     |
| Setup  | Tolerance frames        | <a href="#">0</a>                 | start unbridged, then scan                                       |

| Tab         | Field                  | Value                                   | Why                                       |
|-------------|------------------------|-----------------------------------------|-------------------------------------------|
| Setup       | Cell lengths a b c (Å) | <u>15 15 22</u>                         | enables minimum image                     |
| Setup       | Scan $t^*$ values      | <u>0 2 5 10 20</u>                      | reveals the $t^*$ sensitivity             |
| Setup       | Output folder          | (blank)                                 | writes next to the trajectory             |
| Definitions | Reference mode         | Single atom                             | the shell is centred on one acceptor      |
| Definitions | Reference definition   | <u>295</u>                              | 1-based index                             |
| Definitions | Observed mode          | Each atom in a list, tracked separately | pool statistics over all candidate donors |
| Definitions | Observed definition    | <u>15,16,17,...,294</u>                 | every H in the box                        |

Then press **Run MRT**, and read the summary on the **Results** tab.

**Reading the result.** For this system the tool reports:

```

Average occupancy <h>: 0.00734480
Complete residence events: 272
Censored events (clipped by the start or end of the run): 4

Mean residence time <T>: 95.62500000 frames +/- 64.68269375
IMM survival integral: 658.81120365 frames
Intermittent time from R(t): 1187.16839960 frames
(integration rule: zero_crossing, 1512 of 1512 lags used)

TOLERANCE ( $t^*$ ) SENSITIVITY
-----
----
   $t^*$  (frames)    events          <T>      IMM integral
            0         272      95.62500000    658.81120365
            2         265      87.46415094    696.09164845
            5         255      91.04705882    696.88557485
           10         237      98.56962025    727.93365504
           20         188     116.59574468    724.58083047

```

Four things to take from it:

- $\langle h \rangle \times 280 \approx 2.06$ , so O295 accepts about two hydrogen bonds on average — a sensible number for a carbonyl or ether oxygen, and a good sanity check that the cutoff and the reference atom are right.
- $\langle T \rangle = 95.6$  frames with an uncertainty of  $\pm 64.7$ . The uncertainty is 68 % of the value, so this is an order-of-magnitude statement, not a precise one.
- The  $t^*$  scan moves  $\langle T \rangle$  from 95.6 to 116.6 frames. Quote the scan, not a single number.

- **1512 of 1512 lags used is a warning.** By §9 that means  $R(t)$  never crossed zero, so 1187 frames is a lower bound that grows if you raise the lag ceiling — not a converged time constant. For this trajectory the intermittent value should not be quoted at all;  $\langle T \rangle$  and  $\tau_{IMM}$  are the defensible numbers.

**A common mistake.** If the reference atom is a water oxygen, its own two covalently bonded hydrogens sit at about 0.96 Å and are permanently inside a 2.4 Å shell. They will dominate the statistics and produce enormous residence times. Either choose a reference that has no bonded hydrogens (as atom 295 here does not), or exclude the bonded hydrogens from the observed list.

## Legacy Mean Residence Time

The legacy MRT workflow is retained for compatibility with older analyses.

Inputs include:

- Shell inner radius.
- Shell outer radius.
- Simulation time step.
- Sampling interval.
- Tolerance frames.
- Atom labels to exclude.
- Atom labels at the shell center.
- Element symbol to investigate.
- TRAJEC.xyz trajectory file.

Typical outputs:

- mrt.dat
- mrt\_total.dat
- mrt\_summary.dat

## Autocorrelation Function

Use **Structural > Autocorrelation function** to calculate the velocity autocorrelation function (VAF), the position autocorrelation function (PAF), and an optional power spectrum (PSD) from a CPMD TRAJECTORY file.

Inputs:

- **Starting frame for VAF and PAF** — first trajectory frame to analyze.
- **Stop frame for VAF and PAF** — last trajectory frame to analyze.
- **Number of frames for each ACF**— the maximum correlation lag (window length, in frames).
- **Simulation time step** (atomic units) and **Sampling interval**(frames between stored steps) — used to build the physical time/frequency axes.
- GEOMETRY.xyz file (defines the number of atoms and element symbols).
- TRAJECTORY file (positions and velocities, atomic units).

Default values (whole trajectory):

- When you load the TRAJECTORYfile, any of the three frame fields left blank is filled automatically so that the analysis uses the **entire trajectory**:
  - *Starting frame* defaults to **1** (first frame).
  - *Stop frame* defaults to the **last available frame**.
  - *Number of frames for each ACF* defaults to about **10% of the trajectory length**, capped at **2000**frames (floor 2). Using ~10% keeps roughly 90% of the frames available as time origins for averaging, which gives a statistically reliable estimate while bounding the computational cost and keeping the power spectrum's frequency resolution sensible on long trajectories.
- Any value you type yourself is preserved; only blank fields are auto-filled. Load GEOMETRY.xyz first, then TRAJECTORY, so the atom count is known when the frames are counted.
- If the TRAJECTORY was produced by a **restarted/continued**CPMD run it may contain markers such as <<<<< NEW DATA >>>>>(and blank lines). These are detected and skipped automatically, and the output box reports how many were found. Frames on either side of a restart are treated as one continuous trajectory.

How the functions are defined:

- The VAF/PAF are averaged over **all available time origins**(overlapping windows), includelag 0, and are **normalized so that C(0) = 1**.
- Positions are **mean-subtracted per atom**, so the PAF measures fluctuations about each atom's average position rather than absolute coordinates.

Options:

- Save generated newTRAJEC.xyz (positions converted from Bohr to Angstrom).
- Compute the **power spectrum** (vibrational density of states) from the VAF.

Power-spectrum options (used when the PSD switch is enabled):

- **Mass-weight the spectrum**— when checked, each atom's velocity is weighted by its atomic mass, producing the true vibrational density of states (VDOS) and the file `VDOS.dat`; when unchecked, an equal-weight power spectrum is written to `PSD.dat`. Masses are taken from the elements in `GEOMETRY.xyz`.
- **PSD window** — apodization applied before the FFT to suppress spectral leakage: `Welch` (default), `Hann`, or `None`.
- **Zero-pad x** — zero-padding factor (default `5`); higher values interpolate a smoother spectrum without changing peak positions.
- **Frequency unit** — `cm-1` (default), `THz`, or `Hz`.
- **Partial VDOS groups** — an optional field to overlay *partial* spectra (per element or per atom selection) on the total, which helps **assign vibrational modes** (i.e. see which atoms contribute at which frequency). Enter one or more groups separated by `;`. Each group is a comma/space list of element symbols and/or 1-based atom indices or ranges, for example:
  - `H,O` — one partial for hydrogens together with oxygens;
  - `H; C; O` — three separate partials, one per element;
  - `1-8; 15` — atoms 1–8 as one partial and atom 15 as another.Each partial is written to `PSD <group>.dat` (or `VDOS <group>.dat` when mass-weighting is on) and drawn as a labelled curve on the power-spectrum plot, together with the `Total`. Each curve is normalized to its own maximum so the peak positions of every group are visible. Leave the field blank for the total spectrum only.

The power spectrum is computed by the Wiener–Khinchin route: the VAF is mean-removed, mirrored about  $t=0$ , windowed, zero-padded, Fourier transformed, and squared; the one-sided spectrum is normalized so its strongest peak is 1. The frequency axis is built from the **Simulation time step** and **Sampling interval** you provide, so set those correctly for the frequency scale to be physical.

The PAF, VAF and power-spectrum figures open as **interactivematplotlib** windows (zoom, pan, live cursor coordinates, and a save button in the toolbar), which makes it easy to zoom into spectral peaks and read off frequencies. If the program is run as a packaged executable where an interactive window is unavailable, the figures are shown as static images instead.

### *Power Spectrum of a Solute Only (e.g. an Organic Molecule in a Water Box)*

A common case is a simulation of an organic molecule (solute) in a box of water (solvent) where you want the power spectrum of **only the solute**, without the

solvent contribution. Use the **Partial VDOS groups** field, keeping these points in mind:

- **Do not select by element.** The solute and water usually share elements (H and O), so an element token such as `O` would also pick up every water oxygen. To isolate the solute you must select it by **atom index**.
- **Use the solute's atom-index range/list.** The indices come from the atom order in your `GEOMETRY.xyz` file (the same order as the `TRAJECTORY`). In most solvated setups the solute is written **first**, followed by the water molecules. Open `GEOMETRY.xyz`, count the atoms belonging to the organic molecule, and enter that range. For a solute that is the first 15 atoms, type:

1-15

- You may also give a list or several ranges, e.g. `1-12,14,17` or `1-15,20-22`. If the solute atoms are not contiguous, list their actual indices.
- **Read the result from the partial curve/file.** The group produces a partial spectrum built from only the selected atoms, so water contributes nothing to it. It is saved as `PSD 1-15.dat` (written as `PSD 1 15.dat`) — or `VDOS 1 15.dat` when mass-weighting is enabled — and drawn as the `1-15` curve on the plot. That file/curve is the solute-only power spectrum.
- **Ignore the `Total` curve.** The tool still computes and overlays the `Total` spectrum for the whole system (solute + water); simply use the solute partial and ignore `Total`. Because each curve is normalized to its own maximum, the large water background does not distort the solute spectrum.
- **Tip:** enable **Mass-weight the spectrum** for a physically proper vibrational density of states of the molecule.

Typical outputs:

- `PAF.dat` — columns: lag (frames), normalized PAF.
- `VAF.dat` — columns: lag (frames), normalized VAF.
- `PSD.dat` (or `VDOS.dat` when mass-weighting is on) when the power spectrum is enabled — columns: frequency (in the chosen unit), normalized power.
- `newTRAJEC.xyz` when the trajectory export option is enabled.

## Single Solute Solvent Box

Use **Structural > Single Solute solvent box** to create a box containing one solute and inserted solvent molecules.

Inputs and options:

- Box lattice vectors.
- Target density.
- Maximum insertion attempts.
- van der Waals scaling factor.
- Extra wall padding or minimum distance.
- Solvent XYZ file.
- Solute XYZ file.
- Randomly rotate solvent.
- Calculate density.
- Periodic minimum-image clash detection.
- Include centered solute.

Typical outputs:

- single solute solvent.txt
- single solute box cmass.xyz
- A generated solvent-box XYZ file.

If the target number of solvent molecules cannot be inserted, the summary reports a warning and the number of successful insertions.

## Mixture of Two Solvent Box

Use **Structural > Mixture of two solvent box** to build a mixed-solvent box, optionally with a centered solute.

Inputs and options:

- Solvent A XYZ file.
- Solvent B XYZ file.
- Optional solute XYZ file.
- Mixture composition.
- Box/density and insertion settings.
- Random solvent rotation.
- Density calculation.
- Periodic minimum-image clash detection.
- Insert centered solute.

Outputs include a generated XYZ structure and a TXT summary with composition and insertion statistics.

## Shared-Wall Double Solvent Box

Use **Structural > Shared-wall double solvent box** to build two adjacent solvent regions that share one face. The module is intended for interface models, bilayer-like solvent arrangements, or any setup where two different solvent environments should occupy neighboring slabs in the same simulation box.

The generated structure is a single combined XYZ file. Box 1 occupies the lower z region, from  $z = 0$  to  $z = c1$ , and Box 2 occupies the upper z region, from  $z = c1$  to  $z = c1 + c2$ . The two boxes share the same  $a$  and  $b$  lattice dimensions, but each box has its own z height. No wall atoms or boundary markers are written at the shared interface.

How it works:

- Reads one or two solvent XYZ templates for each box.
- Optionally reads one solute XYZ template for each box.
- Recenters each molecule template, using center of mass when atomic masses are available.
- Estimates the target number of solvent molecules from the requested density, slab volume, solvent composition, and optional solute mass.
- Randomly inserts solvent molecules into Box 1 and then Box 2.
- Optionally randomizes solvent orientation before insertion.
- Keeps atoms inside the assigned z slab and applies the requested wall padding at the outside boundaries.
- Rejects placements that clash using van der Waals radii scaled by the selected vdW factor.
- Uses the full combined box for periodic minimum-image clash detection when that option is enabled.
- Reports insertion statistics and warns if the target number of molecules cannot be reached.

Shared geometry inputs:

- **Shared a b:** two lattice dimensions used by both boxes.
- **Box 1 c:** z height of the lower region.
- **Box 2 c:** z height of the upper region.
- **Periodic minimum-image clash detection:** checks overlaps across periodic boundaries using the full  $a \times b \times (c1 + c2)$  box.

Per-box inputs and options:

- Wall padding.
- Target density in g/cm<sup>3</sup>.
- Solvent A and Solvent B composition percentages.
- Maximum insertion attempts.
- vdW scale.
- Optional random seed.
- Optional solute gap.
- Rotate solvent molecules during insertion.
- Include a centered solute.
- Solvent A, Solvent B, and optional solute XYZ files.

Typical outputs:

- [shared wall box.xyz](#)
- [shared wall box.txt](#)

Example workflow:

- 1 Open **Structural > Shared-wall double solvent box**.
- 2 Enter shared lattice values, for example [30.0 30.0](#), in **Shared a b**.
- 3 Enter [25.0](#) for **Box 1 c** and [25.0](#) for **Box 2 c**. The combined box will have [c = 50.0](#), with the shared interface at [z = 25.0](#).
- 4 For Box 1, set **Density** to [0.95](#), **Solvent A (%)** to [100](#), **Solvent B (%)** to [0](#), and browse for a Box 1 Solvent A XYZ file such as water.
- 5 For Box 2, set **Density** to [0.80](#), **Solvent A (%)** to [100](#), **Solvent B (%)** to [0](#), and browse for a Box 2 Solvent A XYZ file such as an organic solvent.
- 6 Leave **Rotate Box 1** and **Rotate Box 2** enabled unless you need fixed solvent orientations.
- 7 Enable **Box 1 solute** or **Box 2 solute** only if that slab should contain a centered solute, then browse for the corresponding solute XYZ file and set a solute gap if needed.
- 8 Click **Build Shared-Wall Box**.
- 9 Inspect the progress and summary panel. If the summary warns that not all solvent molecules were inserted, try a larger slab, lower density, smaller vdW scale, or more insertion attempts.

## Thermo Tools

### Classical Rate Constant

Use **Thermo > Classical rate constant** to calculate rate constants from classical transition-state-style inputs.

Inputs include numeric parameters such as temperature range and activation-energy-related fields. The tool validates each field and asks for an output directory.

Typical outputs:

- [rate constant vs temperature.dat](#)
- [LnK vs invT.dat](#)

## General Tools

### 3D Molecular Viewer

Use **Tools > 3D Molecular Viewer** to inspect XYZ structures and trajectories.

**Window layout.** The controls are arranged as a fixed header, four tabs, and a status line:

- **Header** (always visible, whichever tab you are on): the XYZ file field with **Browse**, plus the two main actions — **Display Molecule/Trajectory**, which opens the 3D window, and **Save Current Frame XYZ**.
- **Display** tab: projection, atom style, atom and bond scale, maximum bond length, rotation, and atom labels.
- **Frames** tab: frame navigation and trajectory playback.
- **Measure** tab: distances, angles, dihedrals, and atom coordinates.
- **Box & Performance** tab: periodic box settings and playback performance options.
- **Status line** (bottom): reports the last action, such as trajectory loading progress or the value you just applied.

A typical session is: **Browse** for the file, adjust anything you need on the **Display** tab, then click **Display Molecule/Trajectory**. The 3D window opens alongside the control window, so you can keep changing settings while it is open.

**Numeric fields are applied by pressing Enter.** Type a value into a field — maximum bond length, atom or bond scale, box edge lengths, playback delay, frame step, frame number, or a rotation angle — and press Enter to apply it. There are no separate "Set" buttons.

### Display tab

- **Projection:** [Orthographic](#) or [Perspective](#).
- **Atom style:** [Line style](#), [CPK style](#), or [vdW style](#).
- **Atom scale** and **Bond scale:** relative size of the drawn atoms and bonds.
- **Max bond length:** upper distance limit, in Å, for two atoms to be drawn as bonded.
- **Rotation:** rotate the displayed system about X, Y, and Z (described below).
- **Atom numbers** and **Atomic symbols:** label every atom in the 3D view. Both are **off by default**, so the view starts uncluttered — click individual atoms to read their numbers (see the **Measure** tab's **None** type), or turn **Atom numbers** on to label the whole structure at once.
- **Clear selection:** deselect every atom you have clicked, remove any atom numbers you revealed by clicking, and clear the **Measure** tab's field and result.

### Frames tab

- [|<](#), [<<](#), [>>](#), [>|](#) and the slider move through the trajectory. The field to the left of the slider jumps to a specific frame when you press Enter; the counter to the right shows [current / last](#).
- **Play / Pause** animates the trajectory.
- **Loop mode:** [Loop](#) restarts at the beginning, [Once](#) stops at the end, [Rock](#) reverses direction at each end.
- **Frame step:** how many frames each step or playback advance skips. Use [≤](#) and [≥](#), or type a value and press Enter.
- **Delay (s):** seconds between frames during playback. Larger values play more slowly.
- **Auto-zoom to fit each frame:** rescales the view for every frame. Turn it off for a steadier view during playback.

### Measure tab

- The measurement type starts on **None**, which measures nothing: left-clicking an atom simply shows its number on the canvas, and clicking it again hides the number. Use it to find the indices you want before switching to a real measurement.
- Choose **Bond length**, **Bond angle**, **Dihedral angle**, or **Atom coordinates**, then select the atoms **either** by typing their labels into the field **or** by clicking them in the 3D window. Atom indices are 1-based. The value appears in the **Result** field and on the 3D canvas.
- **Measure** computes the value for whatever is currently in the field — use it after typing. When you select atoms by clicking, the measurement runs on its own as soon as enough atoms are selected. Pressing it while the type is **None** asks you to pick a measurement type first.
- **Clear** empties the field, deselects every clicked atom, removes any atom numbers you revealed by clicking, and removes the measurement from the canvas.
- Switching the type clears the marks from the previous mode, so numbers revealed under **None** disappear when you start a measurement, and a half-finished selection disappears when you go back to **None**.

- The **Result** field gives the fully labelled measurement, for example Measurement: d(1,2) = 1.5000 Å. On the 3D canvas only the number is drawn (1.5000, 120.307), so the value stays readable against the structure. Distances are in Å, angles and dihedrals in degrees.

### *Box & Performance tab*

- **Box a, b, c:** orthorhombic periodic box edge lengths in Å. Fill in all three, then press Enter.
- **Box center:** where the displayed box is placed (described below).
- **Show simulation box:** draw or hide the box.
- **Reduce overlays during playback:** skips some overlay drawing to keep long trajectories smooth.
- **Bond mode:** how connectivity is updated during playback (described below).

The saved current frame includes the atoms from the displayed frame and a comment noting the frame number.

### *Selecting atoms by clicking in the 3D view*

What a left-click does depends on the **Measure** tab's type. A click always means a click *without* dragging — dragging still rotates the view.

With the type on **None** (the default), clicking an atom shows its 1-based number next to it in cyan, and clicking the same atom again removes it. Nothing is measured and the **Measure** field stays empty. Click as many atoms as you like; **Clear** removes all the numbers at once.

With any real measurement type selected, you can select atoms directly in the 3D view instead of typing their labels. The atom turns yellow, gets a cyan #1, #2, ... tag showing the order in which you picked it, and its 1-based index is written into the **Measure** tab's field. Typing and clicking are always both available: whatever ends up in the field is what gets measured.

**Click order is the measurement order.** Selecting atoms 4, 7 and 9 in that order for a **Bond angle** gives the field 4, 7, 9, so atom 7 is the vertex — exactly as if you had typed it. The #1/#2/#3 tags on the canvas tell you which atom is which.

Once you have selected as many atoms as the measurement needs (two for a bond length, three for an angle, four for a dihedral, one for atom coordinates), the measurement runs automatically and the result appears in the **Result** field and on the canvas.

Selecting further atoms keeps working without any clearing step:

- Left-click a selected atom to deselect it.
- Left-click a new atom when the selection is already full and the **oldest** pick drops off. For a bond length, this lets you walk a measurement along a chain of atoms one click at a time, getting a new value at every click.
- Switching the measurement type to one that needs fewer atoms keeps your most recent picks.
- **Clear** (Measure tab) or **Clear selection** (Display tab) deselects everything.

Because a click always writes into the field, it will overwrite part of an entry you typed by hand. If you are working from typed labels, click **Measure** before clicking anywhere in the 3D window.

Selections stay attached to the same atoms as you step through or play the trajectory, because atom ordering is constant across frames. Note that **Reduce overlays during playback** (Box & Performance tab) hides all atom labels while it is on, including the index and pick-order labels.

The rotation controls on the **Display** tab let you rotate the displayed molecular system around the X, Y, and Z axes without changing the coordinates stored in the loaded structure or trajectory. Enter rotation increments in degrees, then either press Enter in one of the angle fields or click **Apply angles** to add those increments to the current molecular orientation. Click **Reset rotation** to return the displayed molecular orientation to zero rotation. The  $\leq$  and  $\geq$  buttons beside each axis start continuous rotation in the negative or positive direction; click the same arrow again to stop it, or click another arrow to switch to that axis and direction.

When the periodic box is shown, it rotates with the molecular system. The **Box center** selector on the **Box & Performance** tab controls where the displayed orthorhombic box is placed:

- **Geometric center**: centers the box on the molecular system's geometric center.
- **Bottom at z=0**: centers the box in the a-b plane while placing the bottom face of the box at z = 0.

Bond mode controls how the viewer updates connectivity while stepping through or playing a trajectory:

- **Static first frame**: calculates bonds from frame 0 and reuses that same connectivity for all frames. This is useful when you want stable visual connectivity during normal vibrations or rotations.
- **Dynamic cached**: recalculates bonds for each frame and stores the result for faster revisiting of frames. This is useful for trajectories where bonds may form or break and you still want smooth playback.

- **Dynamic live:** recalculates bonds every time the current frame is rendered, without using cached bond lists. This is useful when you are actively changing bond-length settings or want the freshest possible connectivity during inspection.

To use this feature, open **Tools > 3D Molecular Viewer**, load an XYZ trajectory, then choose the desired option from the **Bond mode** selector on the **Box & Performance** tab. The viewer updates the displayed bonds using the selected mode.

Example: if a trajectory shows two atoms separating during a dissociation event, choose **Dynamic cached** or **Dynamic live**. As you step through the frames, the bond disappears when the atom distance exceeds the current upper bond-length limit. If you choose **Static first frame**, that bond remains visible throughout playback because the viewer keeps the first-frame connectivity.

### *Molecular Design*

The **Design** tab of the 3D Molecular Viewer lets you draw a molecule with the mouse, pre-optimize its geometry, and send the result straight into the 3D viewer to inspect or save as an XYZ file. It is meant for building a starting structure quickly, without leaving the toolkit.

#### *Opening and using the two windows*

The Design tab itself is a launcher. It has two buttons:

- **Open periodic table** — opens the **Periodic Table** window, where you choose the element to draw with.
- **Open design canvas** — opens the **Molecular Design Canvas** window, where you draw the molecule and run the optimization.

Open both and place them side by side. They are ordinary windows: move, resize or close either one at any time. Pressing a launcher button again brings the matching window back to the front rather than opening a second copy, and the canvas window has its own **Periodic table** button for the same purpose.

**Closing a window never loses your work.** The molecule is held by the tool, not by the window, so you can close the canvas, reopen it, and find your structure exactly as you left it. Closing the Molecular Viewer window closes both design windows with it.

### *Selecting elements from the periodic table*

Click any element in the periodic-table window to select it. The table shows all 118 elements in the usual layout, tinted by block (s, p, d and f), with the selected element highlighted in yellow.

The selection takes effect in the canvas window **immediately**— there is nothing to confirm or apply. The current element is shown in three places at once: under the periodic table, above the canvas, and on the Design tab. Carbon is selected when you start.

### *Placing atoms on the canvas*

Click an empty part of the canvas to place an atom of the selected element. The atom is drawn as a coloured circle with its symbol inside, using the same colours as the 3D viewer.

To build a molecule, select an element, place the atoms you need, then select the next element and continue. Clicking an existing atom **does not** place a second atom on top of it.

**Hydrogens are never added for you.** Draw every atom you want in the final structure, hydrogens included.

### *Creating and modifying bonds*

To create a bond, **press the left mouse button on one atom, drag to a second atom, and release**. A dashed guide line follows the pointer while you drag. A new bond is always a single bond.

To change a bond, **click it**. Each click steps to the next bond type:

single -> double -> triple -> resonance -> single

Clicking a bond that is at the end of its cycle returns it to a single bond, so the same click both raises and lowers the bond type and nothing is ever out of reach. The status line names the new type after every click.

Which types are available depends on the two elements involved:

| Bond           | Cycle                                         |
|----------------|-----------------------------------------------|
| C–C, C–N, N–N  | single → double → triple → resonance → single |
| C–O, C–S       | single → double → resonance → single          |
| O–H, C–F, C–Cl | stays single (the status line explains why)   |

The four types are drawn like this:

| Type      | Appearance                                  |
|-----------|---------------------------------------------|
| single    | one solid line                              |
| double    | two parallel solid lines                    |
| triple    | three parallel solid lines                  |
| resonance | one solid line with a dashed line beside it |

**Resonance bonds** are for delocalized bonding: the aromatic bonds of benzene, the two equivalent C–O bonds of a carboxylate, the C–N bond of an amide. A resonance bond counts as one and a half bonds towards an atom's valence and relaxes to a length between a single and a double bond. Draw benzene as six carbons in a ring with six resonance bonds, and it optimizes to six equal 1.40 Å bonds in a flat ring.

To **delete** a bond, right-click it. To delete an atom, right-click the atom — every bond attached to it goes too. **Undo** steps back through the last 50 edits, including **Clear**.

### *Running the pre-optimization*

**Pre-optimize geometry** turns the drawing into a three-dimensional structure. The 2D positions become the starting x and y coordinates, the structure is lifted slightly off the plane, and the geometry is relaxed with a compact force field: harmonic bond stretching with reference lengths from the covalent radii adjusted for each bond type; angle bending at the VSEPR angle for each atom, counting lone pairs, which is what makes water come out bent rather than linear; a twofold torsion term on double and resonance bonds, which keeps alkenes and aromatic rings flat; and a repulsive term that stops distant parts of the molecule from passing through each other.

When the run finishes, a dialog reports the final energy, the gradient norm, the iteration count and any warnings. A run that does not converge is reported with the minimizer's own message, and its partially relaxed structure is still available to send on.

This is a **pre-optimizer**: it produces a reasonable starting geometry for a real CPMD, cp.x or ORCA calculation, not a converged result. The energies it reports are in relative units and mean nothing outside a single run. It is tuned for main-group, mostly organic molecules; geometries with a steric number of six (octahedral) come out distorted.

### *Sending the structure to the 3D viewer*

**Send to 3D viewer** transfers the optimized structure and **opens the 3D viewer window automatically**, bringing it to the front if it is already open. The connectivity you drew is transferred with it, so the bonds you made are the bonds you see. Once it is there, **Save Current Frame XYZ** writes the structure to an XYZ file.

The 3D view draws every bond the same way whatever its type. Bond order shows up in the geometry instead, where it belongs: a resonance C–C bond arrives visibly shorter than a single one.

If the 3D window cannot be opened — a missing OpenGL driver, a blocked window, a start that takes too long — a dialog reports the reason. The structure has still been transferred, so **Display Molecule/Trajectory** and **Save Current Frame XYZ** remain available.

Editing the drawing again discards the previous geometry, so the structure you send is always the one you last optimized.

### *Validation messages*

The structure is checked before every optimization. Some problems stop the run, because there is no sensible geometry to produce; others are only reported.

**These stop the run** (an error dialog names the atom responsible):

| Message                               | What it means                                                                         | What to do                                                                                                                |
|---------------------------------------|---------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| The sketch has no atoms.              | The canvas is empty.                                                                  | Draw something first.                                                                                                     |
| Atom N (X) is not bonded to anything. | A lone atom sits beside a molecule — almost always a stray click.                     | Bond it, or right-click it to delete it.                                                                                  |
| No covalent radius is known for X ... | The element has no radius in the program's tables, so no bond length can be assigned. | Use a different element; the tables cover the main group, the transition metals, the lanthanides and the early actinides. |

**These are reported but do not stop the run:**

| Message                                                    | What it means                                            | What to do                                                                                |
|------------------------------------------------------------|----------------------------------------------------------|-------------------------------------------------------------------------------------------|
| Unusual valence on atom N (X): 5 bonds where 4 is typical. | The atom has more bonds than the element normally forms. | Nothing, if you meant it — unusual valences are chemically real. Otherwise remove a bond. |

| Message                                                              | What it means                                                                                                      | What to do                                                                   |
|----------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------|
| <u>Isolated resonance bond between atom N (X) and atom M (Y) ...</u> | A resonance bond with no other resonance bond at either end. Delocalization over one isolated bond has no meaning. | Give it a resonance neighbour, or click it round to a single or double bond. |
| <u>The sketch contains N disconnected fragments ...</u>              | More than one separate molecule on the canvas.                                                                     | Nothing, if you meant it — they are optimized together in one structure.     |

Other failures are reported in a dialog as they happen: an optimization that hits its iteration limit, or a 3D viewer that cannot open, both name the underlying cause and leave the structure available.

## Energy Plots

Use **Tools > Energy plots** to visualize CPMD or gqteaMD energy files.

For CPMD ENERGY files, plot options include:

- Fictitious and ionic kinetic energy.
- Temperature.
- Kohn-Sham potential energy.
- Kohn-Sham plus ionic kinetic energy.
- Total energy.
- CPU time by step.

X-axis units:

- Steps.
- Femtoseconds.
- Picoseconds.

For gqteaMD energy files, select the appropriate file type and x-axis option in the tool.

### *JSON plot file*

The **Plot type** dropdown offers a third option, **JSON plot file**, which redraws figures a gQTEA analysis tool has already produced — straight from the file, without recomputing anything.

Every analysis tool that shows interactive figures writes such a file next to your data. Select **JSON plot file**, press **Browse**, pick the file, and the message panel lists what it contains:

Loaded 3 figure(s) from the JSON file:

1. Continuous survival  $S(t)$ , origin-averaged - 1512 points [time (fs) vs  $S(t)$ ]
2. Intermittent correlation  $C(t)$  and rescaled  $R(t)$  - 2 curves, 1512 points each [time (fs) vs correlation]
3. Residence event duration distribution - 44 points [event duration (fs) vs count]

Press **Plot** to open them in the interactive viewer (zoom, pan, save).

Because the file describes the figures completely, **all the other plot options are disabled** while this type is selected — the six CPMD plot switches, the simulation time step, the x-axis unit, and the gqteaMD column selectors. Switching back to either energy-file type restores them.

**Expected format.** A JSON array of figures, where each figure is either a single curve

```
{"x": [...], "y": [...], "xlabel": "...", "ylabel": "...",  
  "title": "...", "xlim": [lo, hi], "ylim": [lo, hi]}
```

or several labelled curves drawn together

```
{"series": [{"x": [...], "y": [...], "label": "..."}, ...],  
  "xlabel": "...", "ylabel": "...", "title": "..."}]
```

xlabel, ylabel, title, xlim and ylim are optional. Within one curve, x and y must be lists of finite numbers of equal length.

**Error handling.** A file that cannot be used is reported with a message identifying the figure at fault, so you can fix it without reading the JSON by hand:

| Problem             | Message                                                            |
|---------------------|--------------------------------------------------------------------|
| File does not exist | JSON file not found: <path>                                        |
| Not JSON, or empty  | The file is not valid JSON (line 1, column 1):<br>Expecting value. |

| Problem                                | Message                                                               |
|----------------------------------------|-----------------------------------------------------------------------|
| Top level is an object                 | <u>The JSON file must contain a list of figures (a JSON array)...</u> |
| Empty array                            | <u>The JSON file contains no figures to plot.</u>                     |
| <u>x</u> and <u>y</u> differ in length | <u>Figure 1: 'x' and 'y' must have the same length (3 vs 2).</u>      |
| Neither <u>x/y</u> nor <u>series</u>   | <u>Figure 1 must provide either 'x' and 'y', or 'series'.</u>         |
| A value is not a number                | <u>Figure 1: 'y' must contain only numbers.</u>                       |

## Molecular Axis Alignment

Use **Tools > Molecular Axis Alignment** to align an XYZ molecule along a chosen axis and save a new XYZ file.

Actions:

- **Browse:** select input XYZ.
- **Save As:** choose output path.
- Run the alignment action from the tool window.

## Select Frames

Use **Tools > Select Frames** to extract a frame range from TRAJEC.xyz.

Inputs:

- Starting frame.
- Number of frames to skip between collected frames.
- Stop frame.
- Optional Gaussian settings.
- Charge and multiplicity when Gaussian input generation is enabled.

Outputs:

- selected\_frames.xyz
- Optional Gaussian .gjf files.

## Frame Selection by Interatomic Distance Range

Use **Tools > Frame selection by interatomic distance range** to extract frames where the distance between two atoms falls in a selected interval.

Inputs:

- Minimum distance.
- Maximum distance.
- Two atom labels.
- TRAJEC.xyz.

Typical outputs:

- A selected-frames XYZ file in the input directory.
- A TXT file with selected-frame distances and statistics.
- frame closest to average.xyz.

This tool is useful after bond-length free-energy analysis, for example to select frames near the minimum Helmholtz free energy.

## CPMD Input to XYZ Converter

Use **Tools > CPMD Input to XYZ Converter** to extract coordinates from a CPMD input file and write an XYZ file.

Actions:

- Browse for a CPMD input file.
- Enter an output filename.
- Click **Convert**.

The converter reads CELL values from the &SYSTEM block and atom coordinates from the &ATOMS block.

## SH Geometry Analyzer

Use **Tools > SH Geometry Analyzer** to group surface hopping trajectory frames by electronic state.

Inputs:

- Number of states.
- Root directory containing simulation subfolders.

Each simulation subfolder should contain:

- SH STATE.dat
- TRAJEC.xyz

Outputs:

- stateX.xyz files inside each subfolder.
- Consolidated stateX.xyz files in the root directory.
- sh\_avg\_perc.dat with average state occupancy percentages.

If a subfolder is missing SH STATE.dat or TRAJEC.xyz, the tool reports a warning and skips that subfolder.

### Convert cp.x .pos File to trajec.xyz

Use *\*Tools > Convert .pos file to trajec.xyz* to convert a Quantum ESPRESSO cp.x position trajectory to XYZ.

Inputs:

- cp.x input file, used to read atom labels and nat.
- \*.pos trajectory file.

Output:

- XYZ trajectory file with coordinates converted from Bohr to Angstrom.

### Compute Forces from cp.x .for File

Use *\*Tools > Compute forces from cp.x .for file* to convert a Quantum ESPRESSO force trajectory into a trajectory force file.

Inputs:

- cp.x input file.
- \*.for force trajectory file.

Click **Convert** to generate the converted force output.

## Coordinate Converter

Use **Tools > Coordinate converter** to convert between molecular coordinate file formats.

Controls:

- Input format selection.
- Output format selection.
- Browse for input file.
- Choose output file.
- **Convert.**

Supported formats are determined by the converter UI and parser implementation. The converter writes a new file rather than modifying the original.

## Convert Coordinates from Angstrom to Bohr

Use **Tools > Convert coordinates from A to Bohr** to convert XYZ coordinates from Angstrom to Bohr.

Inputs:

- Input XYZ file.
- Output filename.

Action:

- **Convert.**

## Unit Converter

Use **Tools > Unit Converter** for scalar unit conversion.

Controls:

- Category.
- Value.
- From unit.
- To unit.
- **Convert.**

If the value is blank or a unit is unsupported, the converter shows an error explaining what to fix.

## Configuration

gqteaWinToga does not currently use a persistent user configuration file. Settings are entered directly in each tool window for each run.

Customization points:

- Choose output directories using **Browse**, **Save Dir**, **Save As**, or Save dialogs when available.
- Leave output-name fields blank in some builders to use automatic names such as <xyzname> cp.in, <xyzname> orca.inp, or <xyzname> gqteaMD.toml.
- Toggle optional outputs such as CSV files, plot display, Gaussian input generation, random solvent rotation, density calculation, and minimum-image clash detection.
- Edit generated input files before running external simulation engines, especially pseudopotential names, basis settings, memory, processor counts, and advanced blocks.

Generated files are usually written in the selected output directory or beside the selected input file. Check the status text in each tool window after running a calculation.

## Practical Examples

### Example 1: Find a Bond-Length Free-Energy Minimum

- 1 Open **Geometry > Bond length analysis**.
- 2 Load TRAJEC.xyz.
- 3 Enter:

```
Maximum r: 3.0
Simulation time step: 5
Sampling interval: 1
Temperature: 300
Atom labels: 1 2
Histogram bin width: 0.02
```

- 1 Click **Analyze**.
- 2 Read the minimum straight from the **RESULTS** block in the output box (Lowest free energy: ... at ... Angstrom), or open the generated free energy \*.dat or its plot.
- 3 Use **Tools > Frame selection by interatomic distance range** to extract frames near that distance.

### Example 2: Prepare a CPMD Collision Restart

- 1 Run CPMD for one step and obtain GEOMETRY.xyz.
- 2 Open **Inputs > Collision Input**.
- 3 Select cpmd as the MD engine.
- 4 Enter attacker atom labels, initial velocity, and target coordinates.
- 5 Browse to GEOMETRY.xyz.
- 6 Click **Input Builder**.
- 7 Rename the generated collision geometry as needed for your CPMD restart.
- 8 Restart CPMD with the GEOFILE keyword so the new velocities are used.

### Example 3: Group Surface Hopping Frames by State

- 1 Organize simulation folders under one root directory.
- 2 Ensure each subfolder contains SH STATE.dat and TRAJEC.xyz.
- 3 Open **Tools > SH Geometry Analyzer**.
- 4 Enter the number of states.
- 5 Browse to the root directory.
- 6 Click **Extract Frames**.
- 7 Review stateX.xyz files and sh avg perc.dat.

### Example 4: Convert a cp.x Trajectory

- 1 Open **\*Tools > Convert .pos file to trajec.xyz**.
- 2 Browse to the cp.x input file.
- 3 Browse to the \*.pos file.

- 4 Click **Convert**.
- 5 Use the generated XYZ trajectory in the viewer or analysis tools.

### Example 5: Analyze All Solute Bond Distances

Suppose atoms 1 2 3 4 are the solute and the remaining atoms are solvent. A small trajectory might begin like this:

```
5
frame 1
C 0.0 1.0 0.0
C 0.0 0.0 0.0
C 1.0 0.0 0.0
H 1.0 0.0 1.0
O 0.0 0.5 0.0
5
frame 2
C 0.0 1.1 0.0
C 0.0 0.0 0.0
C 1.1 0.0 0.0
H 1.0 0.0 1.1
O 0.0 0.5 0.0
```

Atom 5 is a solvent atom close to the solute, but it must not contribute to the solute bond analysis.

- 1 Open **Geometry > All bond distance analysis**.
- 2 Browse to the trajectory file.
- 3 Set **Maximum connection distance (Å)** to 1.5 (or leave blank for the 1.7 default).
- 4 Set **Solute atom indices** to (range syntax also works, e.g. 1-4):

```
1 2 3 4
```

- 1 Keep the output filename as:

```
all_bond_analysis_combined.txt
```

- 1 Click **Analyze**.

The single combined file carries the run metadata and one row per pair (identity + first-bonded distance + statistics + occurrence):

```
# gQTEA All Bond Distance Analysis
# frames_used 2
# max_connection_distance 1.5
# atom_scope solute_atoms
# solute_atom_indices 1 2 3 4
# periodic_boundary none
# row atom_i atom_j element_i element_j first_bonded_distance
average variance standard_deviation occurrence_fraction
frames_bonded
      1      1      2      C      C      1.00000000
1.05000000      0.00250000      0.05000000      1.00000000
2
```

Only solute-solute pairs are reported. To apply periodic boundaries, also set **Cell lattices a b c**; the header then records # cell lengths a b c instead of # periodic boundary none.

## Example 6: Analyze All Solute Bond Angles

Use the same idea for all connected solute angles. For a water-like solute with atoms 1 2 3 and a nearby solvent atom 4:

```
4
frame 1
H  0.0  0.0  0.0
O  1.0  0.0  0.0
H  1.0  1.0  0.0
H  1.0  0.5  0.0
4
frame 2
H  0.0  0.0  0.0
O  1.0  0.0  0.0
H  0.5  0.8660254  0.0
H  1.0  0.5  0.0
```

Atom 4 is solvent and should be excluded.

- 1 Open **Geometry > All bond angle analysis**.
- 2 Browse to the trajectory file.
- 3 Set **Maximum connection distance (A)** to 1.1.

#### 4 Set **Solute atom indices** to:

```
1 2 3
```

#### 1 Keep the output filename as:

```
all_angle_analysis_combined.txt
```

#### 1 Click **Analyze**.

The combined file carries the run metadata and one row per triplet (identity + first-present angle + statistics + occurrence):

```
# gQTEA All Bond Angle Analysis
# frames_used 2
# max_connection_distance 1.1
# atom_scope solute_atoms
# solute_atom_indices 1 2 3
# periodic_boundary none
# row atom_i atom_j atom_k element_i element_j element_k
first_present_angle_degrees average_angle_degrees variance_degrees2
standard_deviation_degrees occurrence_fraction frames_present
      1      1      2      3      H      O      H
90.00000000      74.99999995      225.00000163      15.00000005
1.00000000      2
```

The angle is interpreted as atom i-atom j-atom k, so atom 2 is the central atom in this example. To apply periodic boundaries, also set **Cell lattices a b c**; the header then records # cell lengths a b c.

### Example 7: Analyze All Solute Dihedral Angles

For a solute chain 1-2-3-4 in a solvent environment, use **All dihedral angle analysis**.  
Example trajectory:

```
5
frame 1
C 0.0 1.0 0.0
```

```

C  0.0  0.0  0.0
C  1.0  0.0  0.0
H  1.0  0.0  1.0
O  0.0  0.5  0.0
5
frame 2
C  0.0  1.0  0.0
C  0.0  0.0  0.0
C  1.0  0.0  0.0
H  1.0  0.0 -1.0
O  0.0  0.5  0.0

```

Atom 5 is solvent and should not be used to build dihedral paths.

- 1 Open **Geometry > All dihedral angle analysis**.
- 2 Browse to the trajectory file.
- 3 Set **Maximum connection distance (A)** to 1.1.
- 4 Set **Solute atom indices** to:

```
1 2 3 4
```

- 1 Keep the output filename as:

```
all_dihedral_analysis_combined.txt
```

- 1 Click **Analyze**.

The combined file carries the run metadata and one row per quadruplet (identity + first-present dihedral + statistics + occurrence):

```

# gQTEA All Dihedral Angle Analysis
# frames_used 2
# max_connection_distance 1.1
# dihedral_convention signed_degrees_minus180_to_180
# atom_scope solute_atoms
# solute_atom_indices 1 2 3 4
# periodic_boundary none
# row atom_i atom_j atom_k atom_l element_i element_j element_k
element_l first_present_dihedral_degrees average_dihedral_degrees
variance_degrees2 standard_deviation_degrees occurrence_fraction
frames_present

```

|             |            |              |            |               |   |   |   |
|-------------|------------|--------------|------------|---------------|---|---|---|
|             | 1          | 1            | 2          | 3             | 4 | C | C |
| C           | H          | -90.00000000 | 0.00000000 | 8100.00000000 |   |   |   |
| 90.00000000 | 1.00000000 | 2            |            |               |   |   |   |

Dihedrals are reported as signed degrees using the range `(-180, 180]`. To apply periodic boundaries, also set **Cell lattices a b c**; the header then records `# cell lengths a b c`.

## Formulas and Methods (All Bond / Angle / Dihedral tools)

This section documents every equation implemented in the three connectivity-based tools — [allBondAnalysis.py](#), [allAnglesAnalysis.py](#), and [allDihedralAnalysis.py](#). The first four equations are shared by all three; the last two are specific to the angle and dihedral tools.

Notation:  $r_a$  is the Cartesian position of atom  $a$  in a given frame,  $r_{cut}$  is the *maximum connection distance*, and  $L = (a, b, c)$  are the orthorhombic cell lattices.

### 1. Interatomic vector and minimum-image convention (PBC)

The raw vector between two atoms and, when a cell is supplied, its minimum-image replacement (applied component-wise):

$$\Delta r_{ab} = r_a - r_b, \Delta r_{ab} \leftarrow \Delta r_{ab} - L \text{round} \left( \frac{\Delta r_{ab}}{L} \right)$$

- **Represents:** the shortest vector between two atoms under periodic boundary conditions; it lets bonds that span a box face be measured correctly and tolerates unwrapped coordinates.
- **Where/how used:** in the bond tool it is applied inside `_pair_distances`; in the angle and dihedral tools the same operation is the helper `_minimum_image`, reused by `_bonded_adjacency`, `_calculate_angles`, and `_calculate_dihedrals`.

When the `Cell lattices` field is left blank the subtraction is skipped (`cell lengths is None`) and raw vectors are used. Validity requires  $r_{cut} \leq \frac{1}{2} \min(a, b, c)$ , which the tools enforce in `_prepare_run`.

- **References:** M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, 2nd ed., Oxford University Press (2017), §1.6.3; D. Frenkel and B. Smit, *Understanding Molecular Simulation*, 2nd ed., Academic Press (2002).

## 2. Interatomic distance and the connection criterion

$$d_{ab} = \|\Delta r_{ab}\| = \sqrt{\Delta x_{ab}^2 + \Delta y_{ab}^2 + \Delta z_{ab}^2}, \text{ atoms } a, b \text{ are bonded} \Leftrightarrow d_{ab} \leq r_{cut}.$$

- **Represents:** the Euclidean interatomic distance (using the minimum-image vector above) and the per-frame test that decides whether a pair is "bonded".
- **Where/how used:** the distance is the quantity averaged by the **bondtool**; in all three tools the criterion  $d_{ab} \leq r_{cut}$  is re-evaluated **every frame** to build that frame's connectivity graph ([\\_pair\\_distances/](#)[\\_bonded\\_adjacency](#)). An i-j-k angle is present in a frame only when both  $d_{ji} \leq r_{cut}$  and  $d_{jk} \leq r_{cut}$ ; an i-j-k-l dihedral only when all three of  $d_{ij}, d_{jk}, d_{kl} \leq r_{cut}$ .
- **References:** Allen and Tildesley (2017), Ch. 1; standard Euclidean norm.

## 3. Running mean, population variance, and standard deviation (Welford)

For a quantity  $x$  (a distance, angle, or dihedral) accumulated over the  $N$  frames in which the item is present, with  $x_n$  the value in the  $n$ -th such frame:

$$\mu_n = \mu_{n-1} + \frac{x_n - \mu_{n-1}}{n}, M_{2,n} = M_{2,n-1} + (x_n - \mu_{n-1})(x_n - \mu_n), \\ \sigma^2 = \frac{M_{2,N}}{N}, \sigma = \sqrt{\sigma^2}.$$

- **Represents:** the average  $\mu \equiv \bar{\phantom{x}}$ , the **population** variance  $\sigma^2$  (divided by  $N$ , not  $N - 1$ ), and the standard deviation, computed in a single numerically stable streaming pass.
- **Where/how used:** implemented as the masked Welford update [\\_accumulate](#) in the bond tool and inline within [\\_accumulate\\_frame](#) in the angle/dihedral tools; finalized (variance =  $M_2/N$ ,  $\sigma = \sqrt{\sigma^2}$ ) in [\\_finalize](#). Each connected pair/triplet/quadruplet keeps its own  $(N, \mu, M_2)$  accumulator that advances **only in the frames where the item is present**.
- **References:** B. P. Welford, "Note on a Method for Calculating Corrected Sums of Squares and Products", *Technometrics* 4(3), 419–420 (1962); D. E. Knuth, *The Art of Computer Programming*, Vol. 2, 3rd ed., §4.2.2.

## 4. Occurrence (fraction of frames present)

$$f = \frac{N_{\text{present}}}{N_{\text{frames}}}$$

- **Represents:** how persistent a connection is — the fraction of the analysed trajectory in which the pair/triplet/quadruplet is within the cutoff.  $f = 1$  means present in every frame; a smaller value flags a bond that forms or breaks.
- **Where/how used:** reported as [occurrence fraction](#) (with the raw count [frames\\_bonded/frames\\_present](#)) in each combined output file and in the on-

screen

summary;  $N_{present}$  is the Welford count  $N$  and  $N_{frames}$  is `self.num_frames`.

- **References:** standard occupancy/occurrence definition used in hydrogen-bond and contact analysis, e.g. Allen and Tildesley (2017), Ch. 2 (time averages).

## 5. Bond angle (angle tool)

For the triplet i-j-k with central atom  $j$ , using the minimum-image bond vectors  $v_{ji} = r_i - r_j$  and  $v_{jk} = r_k - r_j$ :

$$\theta_{ijk} = \frac{180}{\pi} \arccos \left( \text{clip} \left( \frac{v_{ji} \cdot v_{jk}}{\|v_{ji}\| \|v_{jk}\|}, -1, 1 \right) \right)$$

- **Represents:** the bond angle in degrees, in the range  $[0, 180]$ . The clip to  $[-1, 1]$  guards against tiny floating-point overshoot of the cosine before `arccos`.
- **Where/how used:** the vectorized helper `calculate_angles` computes this for every i-j-k built from the current frame's connectivity; a degenerate case (two atoms coincident, i.e. a zero-length vector) yields `NaN` and that frame is skipped for the affected triplet rather than aborting the run.
- **References:** dot-product angle definition; Allen and Tildesley (2017), §1.3. Numerical clipping of the argument to `arccos` is standard practice.

## 6. Dihedral (torsion) angle (dihedral tool)

For the quadruplet i-j-k-l, using the minimum-image bond vectors  $b_1 = r_j - r_i$ ,  $b_2 = r_k - r_j$ ,  $b_3 = r_l - r_k$ , and the plane normals  $n_1 = b_1 \times b_2$ ,  $n_2 = b_2 \times b_3$ :

$$\phi_{ijkl} = \frac{180}{\pi} \text{atan2} \left( (\hat{n}_1 \times \hat{n}_2) \cdot \hat{b}_2, \hat{n}_1 \cdot \hat{n}_2 \right), \hat{u} \equiv \frac{u}{\|u\|}$$

- **Represents:** the **signed** torsion angle in the range  $(-180, 180]$ . The `atan2` form gives the correct sign (chirality) directly, unlike a bare `arccos` of the normals' dot product.
- **Where/how used:** the vectorized helper `calculate_dihedrals` computes this for every i-j-k-l built around each central bond  $j-k$  in the current frame. A degenerate geometry (overlapping central atoms  $\|b_2\| = 0$ , or a collinear triple giving  $\|n_1\| = 0$  or  $\|n_2\| = 0$ ) yields `NaN` and is skipped for that frame. The convention matches the single **Dihedral angle analysis** tool and is recorded in the output header as `# dihedral convention signed degrees minus180 to 180`.
- **References:** A. Blondel and M. Karplus, "New formulation for derivatives of torsion angles and improper torsion angles in molecular mechanics: Elimination of singularities", *J. Comput. Chem.* **17**(9), 1132–1141 (1996); the `atan2` construction is the standard singularity-free torsion definition (also widely known as the "Praxeolitic" formula).

## Formulas and Methods (Bond Length / Bond Angle / Dihedral Angle tools)

This section documents every equation implemented in the three single-item geometry tools — [bond.py](#) (one atom pair), [bondAngle.py](#) (one i-j-k triplet), and [dihedralAngle.py](#) (one i-j-k-l quadruplet). These tools track one selected item across the whole trajectory, build its distribution, and derive a free-energy profile from it. Everything they share lives in [analysisCommon.py](#), so equations 1, 2, 6, 7, 8, 10, and 11 below are literally the same code in all three tools.

Notation:  $r_a$  is the Cartesian position of atom  $a$  in a given frame,  $L = (a, b, c)$  are the orthorhombic cell lengths,  $x$  stands for whichever coordinate the tool measures ( $r$ ,  $\theta$ , or  $\phi$ ), and  $N$  is the number of frames actually used.

*Note on the sibling section. The [All Bond / Angle / Dihedral tools](#) above share the geometric definitions (equations 3–5 here match equations 2, 5, and 6 there) but differ in two ways: they use a running Welford population variance ( $\div N$ ), whereas these tools use the sample variance ( $\div (N - 1)$ , see equation 7); and they have no free-energy step at all.*

### 1. Physical constants and the simulation time axis

$$\tau_{a.u.} = 0.02418884326505 \text{ fs}, R = 1.987204 \times 10^{-3} \text{ kcal mol}^{-1} \text{ K}^{-1}$$
$$\Delta t_{ps} = \frac{\Delta t_{a.u.} N_{samp} \tau_{a.u.}}{1000}, t_n = n \Delta t_{ps}, n = 0, 1, \dots, N - 1$$

- **Represents:** the conversion from CPMD's internal units to picoseconds.  $\Delta t_{a.u.}$  is the **Simulation Time Step** field and  $N_{samp}$  the **Sampling Interval (frames)** field — the number of MD steps between two stored frames — so their product is the real time between consecutive frames of [TRAJEC.xyz](#). Note that the sampling interval only rescales the time axis; it does not subsample the trajectory.
- **Where/how used:** [GeometryAnalysisBase.time\\_increment\\_ps](#) and the class attributes [atufs](#) and [gas\\_constant](#). The time column of every [.dat/.csv](#) time series is  $t_n$ ;  $R$  appears in equations 8 and 9.
- **References:** CODATA recommended values (atomic unit of time,  $2.4188843265 \times 10^{-17}$  s; molar gas constant  $8.314462618 \text{ J mol}^{-1} \text{ K}^{-1}$ , converted with  $1 \text{ cal} = 4.184 \text{ J}$ ).

## 2. Interatomic vector and minimum-image convention (optional PBC)

$$\Delta r_{ab} = r_a - r_b, \Delta r_{ab} \leftarrow \Delta r_{ab} - L \text{round} \left( \frac{\Delta r_{ab}}{L} \right)$$

- **Represents:** the shortest vector between two atoms under periodic boundary conditions, applied component-wise. It lets an item that spans a box face be measured correctly and tolerates unwrapped coordinates.
- **Where/how used:** [GeometryAnalysisBase.minimum\\_image](#). Applied to the single pair vector in the bond tool, to **both arms**  $v_{ji}, v_{jk}$  in the angle tool, and to **all three** bond vectors  $b_1, b_2, b_3$  in the dihedral tool. When **Cell lengths a b** is left blank the subtraction is skipped ([cell lengths is None](#)) and raw vectors are used.
- **Validity:** the convention is exact only for true separations up to  $\frac{1}{2} \min(a, b, c)$ . The bond tool *enforces* this by requiring  $r_{max} \leq \frac{1}{2} \min(a, b, c)$  in [read\\_params](#). The angle and dihedral tools have no equivalent range parameter to bound, so they instead **warn** when a computed arm or bond exceeds that limit.
- **References:** M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, 2nd ed., Oxford University Press (2017), §1.6.3; D. Frenkel and B. Smit, *Understanding Molecular Simulation*, 2nd ed., Academic Press (2002).

## 3. Bond length (bond tool)

$$r_n = \|\Delta r_{ab}\| = \sqrt{\Delta x_{ab}^2 + \Delta y_{ab}^2 + \Delta z_{ab}^2}$$

- **Represents:** the Euclidean distance in ångström between the two selected atoms in frame  $n$ , using the minimum-image vector of equation 2.
- **Where/how used:** [BondAnalyser.bond\\_length](#), computed for all frames at once as an  $(N, 2, 3)$  array. Written to [bond <pair>.dat](#) as  $(t_n, r_n)$ .
- **References:** standard Euclidean norm; Allen and Tildesley (2017), Ch. 1.

## 4. Bond angle (angle tool)

For the triplet i-j-k with vertex  $j$ , using the minimum-image arms  $v_{ji} = r_i - r_j$  and  $v_{jk} = r_k - r_j$ :

$$\theta_n = \frac{180}{\pi} \arccos \left( \text{clip} \left( \frac{v_{ji} \cdot v_{jk}}{\|v_{ji}\| \|v_{jk}\|}, -1, 1 \right) \right) \in [0, 180]$$

- **Represents:** the bond angle in degrees at the central atom. The clip to  $[-1, 1]$  guards against tiny floating-point overshoot of the cosine before [arccos](#).
- **Where/how used:** [BondAngleAnalyser.bond\\_angle](#), vectorized over all frames.

**Adegenerateframe** — either arm of zero length, i.e. an atom sitting on the vertex — has no defined angle; it is dropped, counted in [skipped degenerate](#), and reported in the summary. The time axis is taken from the original frame index, so dropping a frame does not shift later times.

- **References:** dot-product angle definition; Allen and Tildesley (2017), §1.3. Numerical clipping of the [arccos](#) argument is standard practice.

## 5. Dihedral (torsion) angle and range convention (dihedral tool)

For the quadruplet i-j-k-l, with minimum-image bond vectors  $b_1 = r_j - r_i$ ,  $b_2 = r_k - r_j$ ,  $b_3 = r_l - r_k$  and plane normals  $n_1 = b_1 \times b_2$ ,  $n_2 = b_2 \times b_3$ :

$$\phi_n = \frac{180}{\pi} \operatorname{atan2}((\hat{n}_1 \times \hat{n}_2) \cdot \hat{u}, \hat{n}_1 \cdot \hat{n}_2) \in (-180, 180], \hat{u} \equiv \frac{u}{\|u\|}$$

The reported value then depends on the **Wrap dihedral to [-180, 180]** switch:

$$\phi'_n = \begin{cases} \phi_n, & \text{wrap on} \Rightarrow \phi' \in (-180, 180], \\ \phi_n + 360 \text{ if } \phi_n < 0, \text{ else } \phi_n, & \text{wrap off} \Rightarrow \phi' \in [0, 360). \end{cases}$$

- **Represents:** the **signed** torsion angle. The [atan2](#) form gives the correct sign (chirality) directly, unlike a bare [arccos](#) of the normals' dot product, and is free of the singularity that afflicts the naive formulation.
- **Where/how used:** [DihedralAngleAnalyser.dihedral\\_angle](#), vectorized over all frames. A degenerate geometry —  $\|b_2\| = 0$ , or a collinear triple giving  $\|n_1\| = 0$  or  $\|n_2\| = 0$  — has no dihedral plane; those frames are dropped and counted. **The wrap switch also sets the histogram range of equation 6:**  $[-180, 180]$  when on,  $[0, x_{\max}]$  when off. This coupling matters — a  $[0, x_{\max}]$  histogram applied to signed angles would give every negative angle a negative bin index and silently discard it.
- **References:** A. Blondel and M. Karplus, "New formulation for derivatives of torsion angles and improper torsion angles in molecular mechanics: Elimination of singularities", *J. Comput. Chem.* **17**(9), 1132–1141 (1996); the [atan2](#) construction is the standard singularity-free torsion definition.

## 6. Histogram binning and the probability distribution

With bin width  $\Delta x$  over the range  $[x_{\min}, x_{\max})$ :

$$N_b = \lfloor \frac{x_{\max} - x_{\min}}{\Delta x} \rfloor, k_n = \lfloor \frac{x_n - x_{\min}}{\Delta x} \rfloor, x_k = x_{\min} + \Delta x(k + \frac{1}{2})$$

A sample is retained only when  $0 \leq k_n < N_b$ ; the rest are counted as out-of-range. With  $n_k$  the count in bin  $k$  and  $N_{in} = \sum_k n_k$  the number retained:

$$P_k = \frac{n_k}{N_{in}}, P_k^{\%} = 100 P_k, N_{out} = N - N_{in}$$

- **Represents:** the normalized probability of finding the coordinate in bin  $k$ , reported as a percentage.  $x_k$  is the **bin centre**, which is the abscissa written to the distribution and free-energy files.
- **Where/how used:** `GeometryAnalysisBase.histogram_percent`. The ranges are  $[0, r_{max})$  for the bond tool,  $[0, \theta_{max})$  for the angle tool, and either  $[-180, 180)$  or  $[0, \phi_{max})$  for the dihedral tool (equation 5).
- **Important:**  $P_k$  is normalized over the **retained** samples only, so out-of-range values bias both the distribution and the free energy derived from it. All three tools therefore report  $N_{out}$  in a dialog and in the summary. Out-of-range values are **excluded, never clamped** into the end bin — clamping would silently inflate the last bar.
- **References:** standard frequency-histogram estimator of a probability density; Frenkel and Smit (2002), Ch. 4.

## 7. Sample mean, variance, and standard deviation

$$\bar{x} = \frac{1}{N} \sum_{n=1}^N x_n, s^2 = \frac{1}{N-1} \sum_{n=1}^N (x_n - \bar{x})^2, s = \sqrt{s^2}$$

- **Represents:** the average value of the coordinate over the trajectory and its spread. The  $N - 1$  denominator is **Bessel's correction**, giving the unbiased *sample* variance.
- **Where/how used:** `GeometryAnalysisBase.basic_stats`, implemented as `numpy.var(..., ddof=1) / numpy.std(..., ddof=1)`. Reported in the **RESULTS** block of the summary and in `summary *.txt`. For  $N = 1$  both are defined as 0.
- **Note:** this differs deliberately from the **population** variance ( $\div N$ ) used by the Welford accumulator in the *All* tools — those average over a varying set of frames per item, these over one fixed series.
- **References:** Bessel's correction, standard in any statistics text; the convention matches Python's `statistics.variance` and `statistics.stdev`, which these tools previously used.

## 8. Free energy from the probability distribution

$$G(x_k) = -RT \ln P_k$$

- **Represents:** the free-energy profile along the chosen coordinate, in kcal/mol, obtained by inverting the Boltzmann relation  $P \propto \exp(-G/RT)$ . Only **populated** bins ( $P_k > 0$ ) contribute, since  $\ln 0$  is undefined; empty bins leave gaps in the curve rather than spikes.

- **Where/how used:** [GeometryAnalysisBase.free energy from histogram](#), called by all three tools.  $T$  is the **Simulation Temperature (K)** field. The minimum of the curve and the coordinate at which it occurs are reported in the summary.
- **Important:**  $G$  is defined only up to an **additive constant**, and its absolute value depends on the bin width through  $P_k$ . Compare *shapes* and *barrier heights* between runs, not absolute numbers — and only between runs that used the same bin width.
- **Numerical guard:** probabilities are floored at  $10^{-12}$  before the logarithm to avoid  $-\infty$  from underflow.
- **References:** D. Chandler, *Introduction to Modern Statistical Mechanics*, Oxford University Press (1987), Ch. 7; Frenkel and Smit (2002), Ch. 7; M. E. Tuckerman, *Statistical Mechanics: Theory and Molecular Simulation*, Oxford University Press (2010), Ch. 8. See also J. Mol. Model. **17**, 2159–2168 (2011), DOI 10.1007/s00894-010-0939-6, and *Science* **275**, 817 (1997), DOI 10.1126/science.275.5301.817, the references cited in the tools' own help text.

## 9. Jacobian (volume-element) correction and the potential of mean force

The three coordinates do **not** share the same measure. Their volume elements are

$$dV = 4\pi r^2 dr(\text{distance}), d\Omega \propto \sin \theta d\theta(\text{angle}), d\phi(\text{dihedral, uniform}).$$

Dividing out that geometric weight turns the raw profile of equation 8 into the potential of mean force  $W$ :

$$W(r) = -RT \ln \frac{P(r)}{r^2}, W(\theta) = -RT \ln \frac{P(\theta)}{\sin \theta}, W(\phi) = -RT \ln P(\phi).$$

- **Represents:** the free energy with the *entropic* contribution of the coordinate's own volume element removed. Without it, a distance profile shows an apparent bias toward larger and an angle profile toward  $90^\circ$ , purely because there is more phase space there — not because of any interaction.
- **Where/how used:** the [jacobian](#) argument of [GeometryAnalysisBase.free energy from histogram](#).
  - **bond.py** — the **Jacobian  $r^2$  correction (PMF)** switch, off by default.
  - **bondAngle.py** — the **Use  $\sin(\theta)$  Jacobian** switch, off by default.
  - **dihedralAngle.py** — **no switch, deliberately**. A dihedral has a uniform measure, so there is nothing to divide out and  $-RT \ln P(\phi)$  is *already* the potential of mean force. The summary records this as [Jacobian: not applicable \(uniform measure\)](#).
- **Numerical guard:** the Jacobian factor is floored at  $10^{-12}$ , which matters for  $\sin \theta$  as  $\theta \rightarrow 0^\circ$  or  $180^\circ$ .
- **Relation to  $g(r)$ :** for a distance this is the same construction that links the radial distribution function to the PMF,  $W(r) = -RT \ln g(r)$  with  $P(r) \propto 4\pi r^2 g(r)$  — see the **Radial Distribution Function** tool.

- **References:** Chandler (1987), §7.3; Tuckerman (2010), Ch. 8; D. Trzesniak, A.-P. E. Kunz and W. F. van Gunsteren, "A comparison of methods to compute the potential of mean force", *ChemPhysChem* **8**(1), 162–169 (2007).

## 10. Moving-average smoothing of the probabilities (optional)

With an odd window and edge padding (the first and last values are repeated so the smoothed series keeps its length):

$$\tilde{p}_k = \frac{1}{w} \sum_{m=-(w-1)/2}^{(w-1)/2} p_{k+m}, w = \max(5, \min(11, \lfloor \frac{N_p}{12} \rfloor))$$

where  $N_p$  is the number of populated bins;  $w$  is incremented by one if even, capped at  $N_p$ , and set to 1 (no smoothing) when  $N_p < 5$ .

- **Represents:** a gentle low-pass filter applied to the probabilities **before** the logarithm of equation 8, which tames the noise that sparse bins produce at fine bin widths.
- **Where/how used:** `GeometryAnalysisBase.smooth_series`, enabled by the **Smooth free energy** switch. Default **on** for `bondAngle.py` and `dihedralAngle.py` and **off** for `bond.py`, which preserves each tool's historical output. The summary records the setting as `Smoothed FE`.
- **Caution:** smoothing is cosmetic, not physical. It broadens sharp minima and slightly shifts the reported free-energy minimum, so turn it off when quoting numbers and prefer a coarser bin width to a heavier filter.
- **References:** moving-average (boxcar) filter, standard signal processing; for the trade-off between binning and smoothing in free-energy estimates see Frenkel and Smit (2002), Ch. 7.

## 11. Quantities reported in the summary

Collecting the above, each run ends with:

$$\bar{x}, s^2, s, x_{min}, x_{max}, G_{min} = kG(x_k), x^* = \arg x_k G(x_k)$$

- **Represents:** the average, spread, and extremes of the coordinate, plus the depth and position of the deepest free-energy well.
- **Where/how used:** `GeometryAnalysisBase.render_summary` lays out the `PARAMETERS` and `RESULTS` blocks identically for all three tools; the same text is shown in the output box and written to `summary *.txt`. When the Jacobian is active the label reads `Lowest PMF` instead of `Lowest free energy`.
- **References:** as for equations 7–9.

## Formulas and Methods (Mean Residence Time tool)

This section documents every equation implemented in [meanResidenceTime.py](#), in the order the tool evaluates them. The logic lives in [MeanResidenceTimeCalculator](#); the class is pure NumPy and is exercised headlessly by [tests/test\\_meanResidenceTime.py](#), whose expected values are the analytic results quoted below.

*Note on the two continuous time constants. Equations 5 and 6 define two different quantities — the mean residence time  $\langle T \rangle$  and the mean residual time  $\tau_{IMM}$ . They coincide only for exponential kinetics. Reporting one as though it were the other is the most common error in this analysis, so the tool prints both, separately labelled.*

### Symbols used throughout.

| Symbol          | Meaning                                      |
|-----------------|----------------------------------------------|
| $N$             | number of frames in the trajectory           |
| $M$             | number of observed objects (atoms or groups) |
| $\Delta t$      | time between saved frames                    |
| $r_{cut}$       | cutoff radius defining the shell             |
| $L = (a, b, c)$ | orthorhombic cell edges (optional)           |
| $R(t)$          | position of the reference point in frame $t$ |
| $x_i(t)$        | position of observed object $i$ in frame $t$ |
| $h_i(t)$        | shell indicator (equation 3)                 |
| $t^*$           | tolerance time, in frames                    |
| $n_j$           | length, in frames, of residence event $j$    |

### 1. Reference and observed positions

$$X_{geom}(t) = \frac{1}{n} \sum_{k=1}^n r_k(t), X_{com}(t) = \frac{\sum_{k=1}^n m_k r_k(t)}{\sum_{k=1}^n m_k}$$

- **Represents:** the collapse of a group of  $n$  atoms to a single tracked point per frame — either the unweighted centroid or the mass-weighted centre of mass. For the [single atom](#) and [atom list individual](#) modes the atom's own coordinates are used directly and neither formula is evaluated.
- **Where/how used:** [geometric center series](#) and [center of mass series](#), both vectorized over all frames at once (measured 73× faster than the per-frame loop they replaced, with identical results). Masses come from [AtomicData](#) in [help.py](#) and are looked up **only** for the two centre-of-mass modes, so a purely geometric run never fails on an element missing from the table.
- **References:** standard definitions; see M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, 2nd ed., Oxford University Press (2017), §2.

Atomic masses follow the IUPAC 2021 standard atomic weights as tabulated in [AtomicData](#).

## 2. Separation and the minimum-image convention (optional PBC)

$$d_i(t) = x_i(t) - R(t), d_i(t) \leftarrow d_i(t) - L \text{round} \left( \frac{d_i(t)}{L} \right)$$

$$D_i(t) = \| d_i(t) \| = \sqrt{d_x^2 + d_y^2 + d_z^2}$$

- **Represents:** the separation between the observed object and the reference point. The second expression is the orthorhombic minimum-image convention, applied component-wise: it replaces each separation by the shortest one among all periodic images. It is applied only when cell lengths are supplied.
- **Where/how used:** [build occupancy](#). The convention is exact only while  $r_{cut} \leq \frac{1}{2} \min(a, b, c)$ , and the tool *enforce* that bound, refusing larger cutoffs in both [build occupancy](#) and [collect inputs](#). Coordinates need **not** be wrapped into the box — minimum image tolerates unwrapped trajectories, which is what CPMD normally writes.
- **References:** M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, 2nd ed., Oxford University Press (2017), §1.6.3.

## 3. Occupancy (the shell indicator)

$$h_i(t) = \begin{cases} 1, & D_i(t) \leq r_{cut} \\ 0, & \text{otherwise} \end{cases}$$

- **Represents:** the binary "object  $i$  is inside the shell in frame  $t$ " function. Every quantity below is computed from this  $M \times N$  matrix and nothing else, so the choice of  $r_{cut}$  propagates into all of them.
- **Where/how used:** [build occupancy](#), returned alongside the distances. Take  $r_{cut}$  from the first minimum of the corresponding  $g(r)$ ; the [Radial Distribution Function](#) tool supplies it.
- **References:** R. W. Impey, P. A. Madden and I. R. McDonald, "Hydration and mobility of ions in solution", *J. Phys. Chem.* **87**(25), 5071–5083 (1983).

## 4. Tolerance time $t^*$ (recrossing correction)

$$\dots 1 \ 1 \ \dots g \text{ frames } 1 \ 1 \ \dots \rightarrow \dots 1 \ 1 \ \dots g \text{ frames } 1 \ 1 \ \dots \text{ if } g \leq t^*$$

- **Represents:** bridging short excursions. A gap of  $g$  zero-frames flanked by ones on both sides is overwritten with ones when  $g \leq t^*$ , so a molecule that momentarily jitters back and forth across the cutoff is not counted as having left and returned.
- **Where/how used:** [apply tolerance to occupancy](#), applied to the occupancy matrix before any statistic is taken. Gaps at the very **start or end** of the

trajectory are deliberately never bridged, because there is no evidence the object was inside before or after the run. Because  $t^*$  can only merge events, never split them,  $\langle T \rangle$  is non-decreasing in  $t^*$ .

- **References:** Impey, Madden and McDonald (1983), as above, introduced  $t^*$ . D. Laage and J. T. Hynes, "On the residence time for water in a solute hydration shell", *J. Phys. Chem. B* **112**(26), 7697–7701 (2008), showed the result is *extremely* sensitive to  $t^*$  and that the conventional  $t^* = 2\text{ps}$  can seriously overestimate residence times for low-barrier exchange — which is why the tool provides a  $t^*$  scan rather than a single value.

## 5. Residence events, censoring, and the mean residence time

$$T_j = n_j \Delta t, \langle T \rangle = \frac{1}{N_{ev}} \sum_{j=1}^{N_{ev}} T_j$$

- **Represents:** each maximal run of ones in  $h_i(t)$  is one residence event of duration  $T_j$ ;  $\langle T \rangle$  is their mean. This is the direct residence-time estimator, free of the assumptions built into the correlation-function routes.
- **Where/how used:** runs, extract event durations and count boundary events. Runs touching frame 0 or frame  $N - 1$  are **censored** — excluded from  $\{T_j\}$  and reported separately — because they were still in progress at the boundary and their true duration is unknown. Counting them at their truncated length biases  $\langle T \rangle$  downwards, and does so worst for the longest-lived species. If the censored count is a large fraction of the events, the trajectory is too short for the process being measured.
- **References:** H. R. Sánchez, "Residence Times from Molecular Dynamics Simulations", *J. Phys. Chem. B* **126**(20), 3843–3857 (2022).

## 6. Continuous survival function and the IMM time constant

$$S(l) = \frac{\sum_j \max(0, n_j - l)}{\sum_j n_j}, t = l \Delta t$$

$$\tau_{IMM} = \Delta t \sum_{l=0}^{l_{max}-1} S(l) = \frac{\langle T^2 \rangle}{2 \langle T \rangle}$$

- **Represents:**  $S(l)$  is the probability that an object found inside the shell at an arbitrary **occupied frame** is still *continuously* inside  $l$  frames later. The average runs over time **origins**, not over events, so a long event contributes as many origins as it has frames. This is then  $n_{ion}(t)$  of Impey, Madden and McDonald. Its integral is mathematically the mean *residual* time  $\langle T^2 \rangle / (2 \langle T \rangle)$ , which equals  $\langle T \rangle$  **only** for exponential kinetics and exceeds it whenever

the duration distribution is broad — the usual case for hydration shells.

- **Where/how used:** [compute origin averaged survival](#). The closed form above, summing over run lengths  $n_j$ , is exactly equivalent to scanning every time origin explicitly (a unit test checks it against a brute-force origin scan) but costs  $O(events)$  per lag instead of  $O(N^2)$ .  
For a set of uniform runs of  $n$  frames the integral is exactly  $\frac{1}{2}(n+1)\Delta t$ , which is the analytic value the tests assert.
- **References:** Impey, Madden and McDonald (1983); A. E. García and L. Stiller, "Computation of the mean residence time of water in the hydration shells of biomolecules", *J. Comput. Chem.* **14**(11), 1396–1406 (1993); B. Halle and F. Persson, "Analysis of Protein Dynamics Simulations by a Stochastic Point Process Approach", *J. Chem. Theory Comput.* **9**(6), 2838–2848 (2013), for computing the survival function from the residence-time sequence rather than by averaging over initial times; S. Mukherjee, S. Mondal and B. Bagchi, *J. Chem. Phys.* **147**, 024901 (2017), for the broad, log-normal duration distributions that make the two time constants diverge.

## 7. Intermittent correlation function

$$C(l) = \frac{\sum_{i=1}^M \sum_{t_0=0}^{N-l-1} h_i(t_0) h_i(t_0 + l)}{\sum_{i=1}^M \sum_{t_0=0}^{N-l-1} h_i(t_0)}, \langle h \rangle = \frac{1}{MN} \sum_{i=1}^M \sum_{t=0}^{N-1} h_i(t)$$

$$R(l) = \frac{C(l) - \langle h \rangle}{1 - \langle h \rangle}$$

- **Represents:**  $C(l)$  is the probability that an object inside the shell at  $t_0$  is inside again at  $t_0 + l$ , **whether or not it left in between**. It therefore probes exchange rather than uninterrupted residence, and is the natural partner to equation 6. Since  $C(l) \rightarrow \langle h \rangle$  at long lag,  $R(l)$  is the rescaling running from  $R(0) = 1$  to  $R(\infty) = 0$ , which is what can be integrated to a time constant.
- **Where/how used:** [compute intermittent functions](#). Note the asymmetry:  $C(l)$  is normalized over the origins available *at that lag*, while  $\langle h \rangle$  is the global mean occupancy over all frames, so the two denominators differ slightly. When  $\langle h \rangle = 1$  the object never leaves and  $R$  is defined as 1 for all lags.  $C$  is clipped to  $[0,1]$  and  $R$  to  $[-1,1]$  to absorb floating-point overshoot.
- **References:** Impey, Madden and McDonald (1983). The same intermittent/continuous pairing underlies the hydrogen-bond kinetics of A. Luzar and D. Chandler, "Hydrogen-bond kinetics in liquid water", *Nature* **379**, 55–57 (1996).

## 8. Lag ceiling

$$l_{max} = \max(1, \min(N, \lfloor \frac{N}{10} \rfloor))$$

- **Represents:** the largest lag at which the correlation functions are evaluated, when the **Maximum lag** field is left blank. At lag  $l$  only  $N - l$  time origins are available, so the longest lags average over a handful of samples and are noisy; integrating them into a residence time is meaningless.
- **Where/how used:** [default max lag](#), applied to both equation 6 and equation 7. It can be overridden from the **Advanced** group on the Setup tab, which is how you check whether a reported time constant has converged.
- **References:** Halle and Persson (2013), for the growth of statistical error with lag in this class of correlation function.

## 9. Turning $R(t)$ into a time constant

$$\text{zero crossing: } \tau = \Delta t \sum_{l=0}^{l_0-1} R(l), l_0 = \min\{l: R(l) < 0\}$$

$$\text{exponential: } \ln R(l) = A - \frac{l \Delta t}{\tau} \text{ (least squares over } \{l: R(l) > 0\}), \tau = -\frac{1}{\text{slope}}$$

$$\text{full: } \tau = \Delta t \sum_{l=0}^{l_{max}-1} R(l)$$

- **Represents:** three ways to reduce the decaying correlation to a single time. Zero crossing integrates up to the point the curve first goes negative; the exponential rule assumes single-exponential decay and reports the fitted  $\tau$ ; full integrates the whole evaluated range.
- **Where/how used:** [integrate correlation](#), selected by the **Integration rule** control. The exponential fit requires at least three lags with  $R(l) > 0$  and falls back to the zero-crossing rule if the fitted slope is not negative. The summary always records which rule was applied and how many lags it covered, so a run is reproducible from its own output.
- **Convergence check:** if the number of lags used **equals**  $l_{max}$ , then  $R(t)$  never crossed zero, and  $\tau$  is a lower bound that grows with  $l_{max}$  rather than a converged time constant. Raise the lag ceiling and see whether the value is stable. If it keeps growing, the trajectory is too short to resolve the process and the intermittent time constant should not be quoted at all — report  $\langle T \rangle$  and  $\tau_{IMM}$  instead.
- **References:** the exponential form is the standard single-relaxation-time reading of the intermittent correlation function used by Impey, Madden and McDonald (1983) and, for hydrogen-bond kinetics, by A. Luzar and D. Chandler, *Nature* **379**, 55–57 (1996). H. R. Sánchez, *J. Phys. Chem. B* **126**, 3843 (2022), analyses the assumptions this step introduces relative to the direct estimator of equation 5.

## 10. Uncertainty on the mean residence time

$$\sigma(\langle T \rangle) = \frac{1}{\sqrt{B}} \sqrt{\frac{1}{B-1} \sum_{b=1}^B (\bar{c}_b - \bar{c})^2}, B = 5$$

- **Represents:** the standard error of  $\langle T \rangle$ , estimated from the means  $\bar{c}_b$  of  $B$  contiguous blocks of the event-duration series.
- **Where/how used:** [block average uncertainty](#). Block averaging is used rather than a plain standard error because successive residence events are **not** independent samples; a naive  $s/\sqrt{n}$  would understate the error. Returns NaN when there is not enough data to form two blocks. Only  $\langle T \rangle$  carries an uncertainty —  $\tau_{IMM}$  and the intermittent time constant do not.
- **References:** H. R. Sánchez et al., "Evaluating the uncertainty in mean residual times: Estimators based on residence times from discrete time processes", *Commun. Nonlinear Sci. Numer. Simul.* (2024).

## Troubleshooting

### "No file was selected!"

You closed a file dialog without choosing a file. Click **Browse** again and select the required input.

### "Please input a valid value for ..."

One of the numeric fields is blank or contains text that cannot be converted to a number. Check the field named in the message and enter a valid value.

### "Invalid format for ... Please input exactly two atom labels."

The atom-label field has the wrong number of labels. Bond analysis needs two labels, bond angle needs three, and dihedral angle needs four. Separate labels with spaces, not commas.

### "Failed to open file" or "Failed to read file"

The selected file could not be opened or parsed. Confirm that:

- The file still exists.
- You have permission to read it.

- The file format matches the selected tool.
- XYZ files begin with the atom count and contain valid coordinate lines.

### **"The file is empty!"**

The selected file has no readable content. Choose a valid geometry, trajectory, or input file.

### **"Invalid line format in TRAJECTORY file"**

The trajectory file does not match the expected numeric layout. Check that the file came from the expected simulation program and has not been truncated or edited incorrectly.

### **"The stop frame must be less than the total number of frames"**

The selected stop frame is outside the trajectory. Use the frame count displayed after browsing for the trajectory, then choose a smaller stop frame.

### **ENERGY file format errors**

The CPMD energy plotter expects a valid ENERGY file with eight numeric columns. If plotting fails, verify that the file is complete and that you selected the correct plot type.

### **Solvent box insertion warnings**

If the solvent box builder cannot insert all requested molecules, the target density or minimum-distance settings may be too restrictive. Try one or more of the following:

- Increase the box dimensions.
- Lower the target density.
- Reduce the minimum distance or van der Waals scaling factor.
- Increase the maximum number of insertion attempts.

### **Missing SH STATE.dat or TRAJEC.xyz**

The SH Geometry Analyzer skips subfolders that do not contain both files. Add the missing files or remove incomplete subfolders from the selected root directory.

## Plot windows do not appear

A tool reports that it finished, but no plot window opens and no error is shown. The interactive plot viewer runs in a separate process and pins matplotlib's `TkAgg` backend, so this almost always means **tkinter is missing**. It fails silently rather than raising.

Check for it:

```
python -c "import tkinter; print(tkinter.TkVersion)"
```

If that command fails, re-run the python.org installer, choose **Modify**, and enable the **tk and IDLE** component. `tkinter` is a standard-library module and cannot be installed with `pip`. See [System requirements that pip cannot install](#).

In a packaged `.exe` build, the same symptom means the `TkAgg` backend was not bundled. See the packaging notes in [readme.md](#).

## Molecular viewer does not open or shows no 3D view

The viewer depends on OpenGL and GLFW. Work through these in order:

- 1 Confirm the packages are installed:

```
python -c "import glfw, OpenGL; print('ok')"
```

- 2 If this fails, reinstall with `python -m pip install -r installation\_requirements.txt`. The required `glfw3.dll` and `freeglut` libraries are bundled inside those wheels, so they do not need to be downloaded separately.
- 3 Confirm your graphics driver provides OpenGL in the current desktop session. Install the vendor driver for your hardware (Intel, NVIDIA or AMD); the Microsoft Basic Display Adapter driver does not include an OpenGL runtime.
- 4 The viewer cannot run over a plain SSH session or in a headless container. A Remote Desktop session may work through a software renderer, but slowly.

If the window still does not open, the failure reason is reported in a dialog and in the status line at the bottom of the Molecular Viewer window.

## Packaged Windows executable cannot find GLFW

When packaging with [auto-py-to-exe](#), include the GLFW DLL manually if needed. It is usually located at a path similar to:

```
venv\Lib\site-packages\glfw\glfw3.dll
```

## Version, Contributors, and License

Version shown by the application:

```
gQTEA-0.4.0 Molecular Analysis Toolkit
```

The launcher source notes that the program was revised and updated in March 2026.

Core development team:

- Ademir J. Camargo - [ajc@ueg.br](mailto:ajc@ueg.br)
- Valter H. C. Silva - [fatioleg@ueg.br](mailto:fatioleg@ueg.br)
- Solemar S. Oliveira - [solemar@ueg.br](mailto:solemar@ueg.br)
- Hamilton B. Napolitano - [hamilton@ueg.br](mailto:hamilton@ueg.br)
- Luciano Ribeiro - [lribeiro@ueg.br](mailto:lribeiro@ueg.br)
- Flavio O. Sanches - [flavio.neto@ifg.edu.br](mailto:flavio.neto@ifg.edu.br)

No license file is currently included in the source tree. Before distributing the program publicly, add a [LICENSE](#) file so users know how they may use, modify, and redistribute the code.