# COSMIC–ASCEC

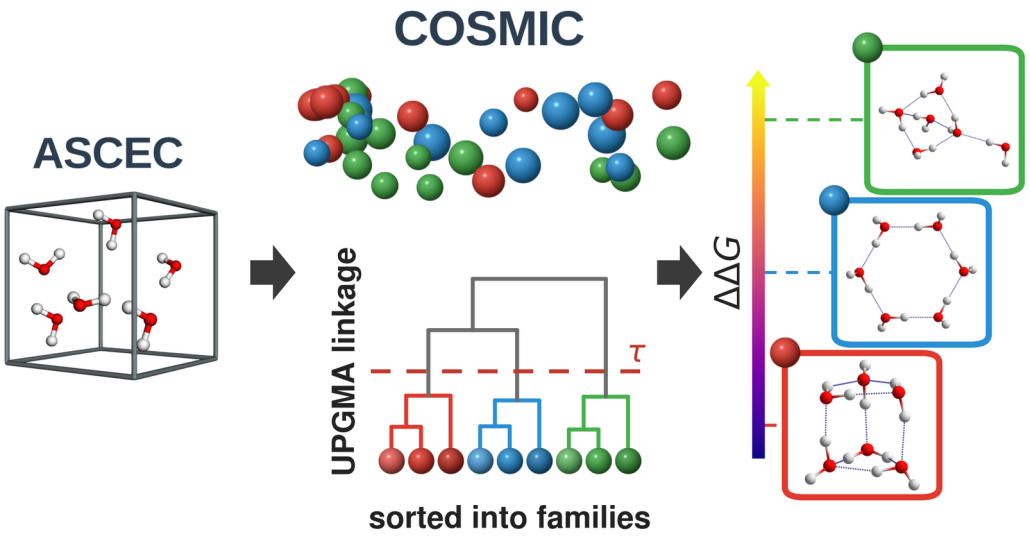
*Automated Configurational Sampling and Topological Screening of Molecular Clusters*

# User Manual

**Manuel Gómez**[†] – **Sara Gómez**[‡] – **Albeiro Restrepo**[†]

Química Física Teórica

[†]*Instituto de Química, Universidad de Antioquia UdeA, Medellín, Colombia*
[‡]*Universidad Nacional de Colombia, Departamento de Química, Bogotá, Colombia*




**Last Updated:** June 2026

# Contents

# 1 Distribution

The ASCEC-v04 & COSMIC-v01 package is distributed via a public GitHub repository:

- **Repository:** https://github.com/manuel2gl/qft-cosmic-ascec
- **License:** GNU General Public License (see LICENSE file for details)
- **Supported OS:** Primarily developed and tested on Linux (Ubuntu, CentOS, Fedora). Preliminary runs are also tested on Windows using xTB 6.7.1. macOS should work with minor adjustments.
- **Contents:**
  - Python scripts for configurational / conformational sampling, clustering, and analysis
  - Documentation and brief theoretical description
  - Example input files and workflows
  - Web based input generator:
    - https://manuel2gl.github.io/qft-cosmic-ascec/
- **Requirements:**
  - Python 3.9 or higher (3.11 suggested)
  - ORCA or xTB for quantum chemistry calculations

# 2 Introduction

## 2.1 What is ASCEC?

**ASCEC** (Annealing Simulado Con Energía Cuántica / Simulated Annealing with Quantum Energy) is a computational tool for automated configurational sampling and analysis of molecular clusters. It integrates:

- **Simulated Annealing**: Efficient exploration of potential energy surfaces
- **Hierarchical Clustering (COSMIC)**: Identification of representative structures using a physicochemical feature vector.

## 2.2 How It Works (Overview)

The ASCEC pipeline operates in three stages:

1. **Sampling (Annealing):** A simulated annealing algorithm randomly moves, rotates, and deforms the molecules inside a box. At each step, a quantum chemistry package (ORCA or xTB) evaluates the energy. The algorithm gradually "cools down", favouring low energy structures while still allowing occasional uphill moves to escape local traps. This produces hundreds of candidate configurations.
2. **Clustering (COSMIC):** Most of these candidates are redundant; to overcome this limitation, the COSMIC (COnfigurational Similarity via Motif Identification Clustering) module compares them using a fingerprint constructed from their energies, orbital vacancies, dipole moments, and other properties. Structures with similar fingerprints are grouped into families. Only one representative per family (a **motif**) is retained for further study.
3. **Refinement:** The motifs are refined at a higher level of theory (e.g., DFT) to obtain publication quality geometries, energies, and thermodynamic populations.

# 3 Theoretical Background

## 3.1 Simulated Annealing Algorithm

The ASCEC module drives the exploration of the configurational space through a Markov Chain Monte Carlo (MCMC) protocol implemented within a simulated annealing framework [1]. To ensure statistical significance, the algorithm generates thousands of trial configurations. The quantum energy of every generated structure is calculated using efficient methods (supporting any theoretical level available in the external package, e.g., GFN2-xTB, PM3, AM1, HF).

The acceptance of a structural perturbation depends on the change in potential energy, $\Delta E = E_{new} - E_{old}$. If the structural change lowers the energy ($\Delta E < 0$), the move is accepted immediately. In the case of an energy increase ($\Delta E > 0$), the acceptance behavior depends on the chosen criterion:

**Standard Metropolis Criterion [2] (Optional)** When explicitly requested (via `--standard` flag), the algorithm follows the classical Boltzmann probability. An uphill move is accepted if a random number $\xi \in [0, 1]$ satisfies:

$$\xi < e^{-\frac{\Delta E}{k_B T}} \tag{1}$$

**Modified Metropolis Criterion (Default)** To mitigate kinetic trapping in local basins and prevent jumps to physically unrealistic high-energy configurations ASCEC implements a modified acceptance condition [3]. Unlike the standard probabilistic approach, which compares the Boltzmann factor to a random number, this method evaluates the relative energy cost of the perturbation. The new structure is accepted if:

$$\Phi(\Delta E) < P(\Delta E) \tag{2}$$

where $P(\Delta E) = e^{-\Delta E/k_B T}$ is the temperature dependent Boltzmann probability, and $\Phi(\Delta E)$ represents the relative energy change defined as:

$$\Phi(\Delta E) = \frac{\Delta E}{|E_{new}|} \tag{3}$$

This criterion was found to be more adequate for this class of problems. By removing the stochastic element found in the standard Metropolis method, the algorithm avoids the accidental rejection of chemically viable structures due to random number generation, while still filtering out moves where the relative energy increase is drastic.

**Temperature Schedules:** The simulation temperature decreases over $n_T$ steps to gradually reduce the acceptance of high-energy states. A choice between a linear (decreasing $T$ by a constant amount) and a geometric (decreasing $T$ by a constant percentage) route is available:

- *Linear Quenching:* The temperature is reduced by a fixed amount $dT$ at each step:

$$T_{n+1} = T_n - dT$$

- *Geometric Quenching:* The temperature is scaled down by a cooling factor $f$ (in %):

$$T_{n+1} = T_n \times \left(1 - \frac{f}{100}\right)$$

### 3.2 Monte Carlo Moves

The configuration space is explored using three perturbation motions:

1. **Translation:** Random displacement of the center-of-mass.
2. **Rotation:** Random rotation around principal axes.
3. **Conformational (Optional):** Modification of internal dihedral angles.

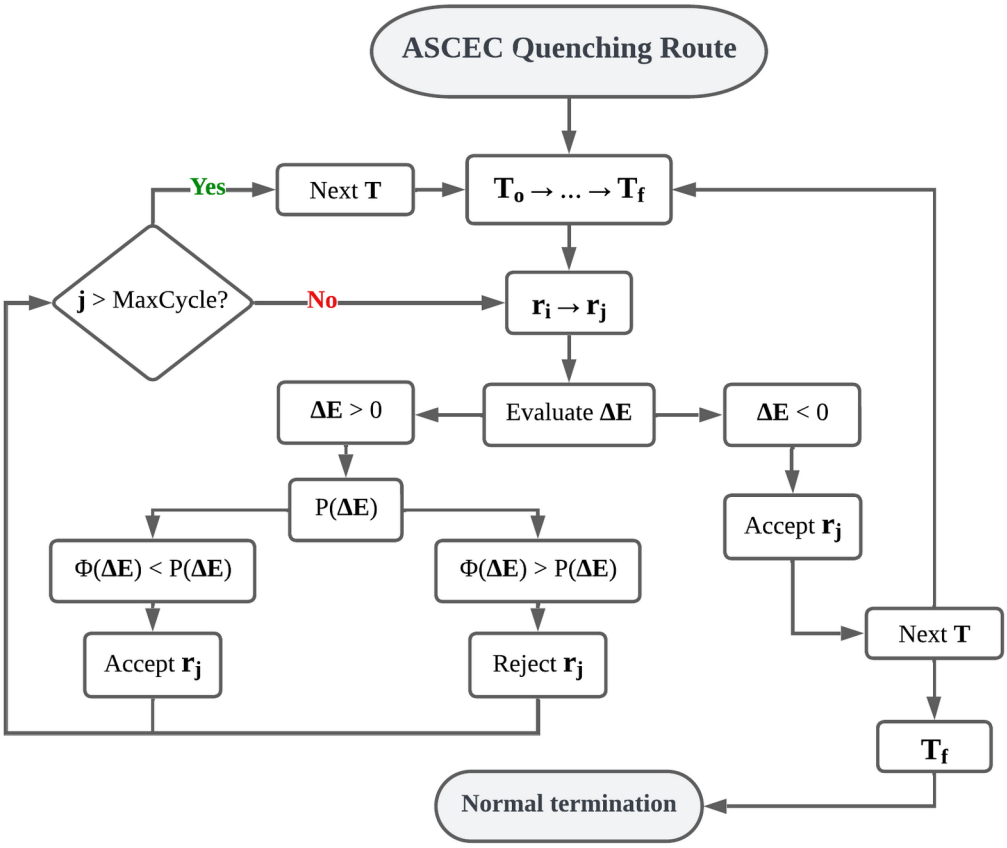



Figure 1: SA procedure using a modified Metropolis criterion.

### 3.3 Topological Clustering (COSMIC)

While stochastic exploration (ASCEC) is effective at traversing energy barriers, it inherently suffers from a data management limitation: the generation of highly redundant datasets. A typical annealing run yields hundreds of candidate structures, the vast majority of which converge to identical basins of attraction. To address this, the **COSMIC** module functions as a post-processing discriminator designed to filter, validate, and cluster the raw output into unique chemical motifs.

The scope of this algorithm is to bridge the gap between stochastic chaos and ordered chemical insight. By reducing massive datasets into non-redundant families, it facilitates the manual revision of the Potential Energy Surface (PES) and ensures that only topologically distinct minima are subjected to high-level quantum mechanical refinement.

### 3.3.1 The Physicochemical Feature Vector

Standard clustering methods often rely solely on geometric superposition (RMSD), which depends on atomic indexing [4]. To overcome this, we define the configurational identity of a structure $i$ using a multi-dimensional scalar feature vector, $\mathbf{x}_i$. This vector aggregates continuous physicochemical descriptors derived from the quantum mechanical data. The **full vector** (up to 15 dimensions) is:

$$\mathbf{x}_i = \left(E_{elec}, V_{nuc}, G, E_{\mathrm{HOMO}}, \Delta E_{\mathrm{HL}}, \mu, A, B, C, \nu_{\mathrm{smallest}}, \nu_{\mathrm{largest}}, n_{\mathrm{HB}}, \overline{d_{\mathrm{HB}}}, \overline{\theta_{\mathrm{HB}}}, \sigma_{d_{\mathrm{HB}}}\right) \tag{4}$$

where the components represent energetics ($E_{elec}$, $G$); $V_{nuc}$: nuclear repulsion energy; $E_{\mathrm{HOMO}}$: energy of the HOMO; $\Delta E_{\mathrm{HL}}$: HOMO–LUMO gap; $\mu$ dipole moment; $A, B, C$: rotational constants; $\nu_{\mathrm{smallest}}, \nu_{\mathrm{largest}}$: smallest and largest vibrational frequencies; $n_{\mathrm{HB}}$: number of hydrogen bonds, and hydrogen bonding descriptors $\overline{d_{\mathrm{HB}}}, \overline{\theta_{\mathrm{HB}}}, \sigma_{d_{\mathrm{HB}}}$: geometrical parameters defining hydrogen bonds within each structure.

All structures are clustered together in a single pool by default no pre-grouping by hydrogen-bond count is performed. H-bond information is encoded indirectly through $\overline{d_{\mathrm{HB}}}$ and $\overline{\theta_{\mathrm{HB}}}$, allowing the clustering algorithm to account for H-bond patterns without imposing rigid partitions. The optional `--group-hb` flag restores per H-bond count grouping.

### 3.3.2 Weighted COSMIC Metric

Since the vector components possess disparate physical units, a feature-wise Z-score standardization is applied $z_i^k = (\gamma_i^k - \bar{\gamma}^k/\sigma^k)$, $\quad Z_i^k = w^k \times z_i^k$. The distance between two structures is then quantified using the Euclidean distance:

$$d\left(\mathbf{X}_i', \mathbf{X}_j'\right) = \sqrt{\left(\mathbf{X}_i' - \mathbf{X}_j'\right) \cdot \left(\mathbf{X}_i' - \mathbf{X}_j'\right)} = \sqrt{\sum_{k=1}^{n_d} \left(Z_i^k - Z_j^k\right)^2} \tag{5}$$

where $w^k$ represents user-definable weights used to bias the clustering focus (e.g., prioritizing H-bonding patterns over soft vibrational modes).

### 3.3.3 Hierarchical Two-Step Clustering

To ensure both computational efficiency and high structural fidelity, the clustering process operates via a hierarchical "funnel" protocol involving two distinct steps:

1. **Step I: Physicochemical Clustering (Coarse-Grained).**
   The algorithm first performs agglomerative hierarchical clustering using UPGMA (average linkage) based on the $z$-score-standardised feature vectors $\mathbf{x}$. This rapidly groups configurations that share similar energetics, electronic states, and general shapes into broad families. The dendrogram is cut at a user-defined distance threshold $\tau$ to obtain a flat partition. This step drastically reduces the dataset size by merging redundant snapshots without performing expensive geometric alignments.

2. **Step II: Optional RMSD Refinement (Fine-Grained).**
   Optionally, a secondary geometric check is applied within each identified cluster. A Root Mean Square Deviation (RMSD) analysis [5] is performed to distinguish stereoisomers or iso-energetic conformers that map to identical scalar vectors (e.g., enantiomers or distinct proton ordering networks with identical energies). This ensures that subtle geometric variations are not lost during the vector-based reduction.

#### 3.3.4 Quality Control

Beyond clustering, the module serves as a topological filter. It screens for the presence of **one or more imaginary frequencies**, indiscriminately identifying first-order saddle points (transition states) and higher-order instabilities. If such a structure clusters tightly with a true minimum, it is discarded as a redundant, non-converged distortion. However, if it forms an isolated cluster, it is flagged as a unique topological feature and output for re-optimization, ensuring that potential basins are not overlooked merely due to incomplete convergence.

# 4 Installation

## 4.1 System Recommendations

Although Python scripts are lightweight, quantum chemistry backends are resource-intensive:

- **RAM:** Minimum 8GB (16GB+ recommended for DFT calculations).
- **CPU:** Multi-core processor (ASCEC can parallelize ORCA jobs).
- **Storage:** SSD recommended. Optimization trajectories can generate large text files.
- **OS:** Linux (Ubuntu/CentOS) is the primary supported environment.
  Windows is supported but may require path adjustments.

## 4.2 Required Software

### 4.2.1 Core Python Dependencies

The core functionality is based on standard Python scientific libraries and other software.

| Package | Purpose |
|---|---|
| **NumPy** | High-performance array operations and linear algebra. |
| **SciPy** | Optimization routines and statistical functions. |
| **scikit-learn** | Algorithms for hierarchical clustering and data preprocessing. |
| **Matplotlib** | Generation of energy plots and dendrograms. |
| **cclib** | Parsing of legacy ORCA ($\leq$ 5.0.x) logs. |
| **OPI** | Parsing of modern ORCA ($\geq$ 6.1) outputs. |
| **OpenBabel** | File generation and format conversion. |
| **xTB** | Highly efficient semiempirical quantum chemical framework. |

Table 1: Python libraries required for ASCEC and COSMIC modules.

***Compatibility Note:** The **COSMIC** module automatically selects the appropriate parser based on the output file version. `cclib` is used for ORCA $\leq$ 5.0.x, while `OPI` is required for ORCA 6.1+. *Note: ORCA 6.0 is `not` supported by either parser; please use v5.0.x or upgrade to v6.1+.* At least one parser must be installed.

#### 4.2.2 Quantum Chemistry Backends

The workflow requires an external electronic structure package to perform QM calculations.

**ORCA:** ORCA [6] is free for academic use and supports a broad range of high-level ab initio and DFT methods. It is fully integrated with both the ASCEC and COSMIC modules. See Section 16 for installation instructions.

**xTB:** xTB [7] is the default backend for both the annealing and preliminary preoptimization stages due to its speed. It is installed automatically from `conda-forge` by `install.sh` and `win_install.bat`.

### 4.3 Step-by-Step Installation

#### 4.3.1 Automatic Installation

*We provide a simplified shell script that automates the entire process. It installs Miniconda (if not present), sets up the dependencies (xTB, OpenBabel, cclib, OPI), and configures the necessary aliases. The aliases point directly to the environment's Python binary, so no manual* `conda activate` *is needed to run* `ascec` *or* `cosmic`*. First download the install.sh script.*

```
wget https://raw.githubusercontent.com/manuel2gl/qft-cosmic-ascec/main/install.sh
```

**Installation Steps**
The installation can be performed by executing the `install.sh` script, which automatically uses a Python 3.11 environment (py11).

```
bash install.sh
```

```
source ~/.bashrc
```

Alternatively, if the `base` environment is preferred, install the latest Miniconda and downgrade the Python version within `base`, then run the script again.

```
micro install.sh
```

```
#=========================================
# CONFIGURATION
#=========================================
# Set to TRUE to create a separate 'py11' environment with Python 3.11
# Set to FALSE to install into the base conda environment (default)

INSTALL_PY11=TRUE
```

*Save the file (Ctrl+S) and exit (Ctrl+Q).*

**Uninstall (Linux)**
To reverse the installation without touching your Conda base, run the provided uninstaller.

```
bash uninstall.sh
```

This removes the `py11` environment, the `~/software/ascec04` directory, and the `ascec`/`cosmic` aliases from `.bashrc`. Conda itself is left intact.

**Windows Installation**
On Windows, the same one-click workflow is provided as a `.bat` file (a hybrid batch/PowerShell script). No prior setup is required — simply double-click `win_install.bat`, or run it from a Command Prompt. The script installs Miniconda if absent, creates the `py11` environment, installs all dependencies (including xTB 6.7.1), and registers `ascec` and `cosmic` as system-wide launchers. If Windows blocks execution, right-click the file and select *Run as administrator*.

```
wget https://raw.githubusercontent.com/manuel2gl/qft-cosmic-ascec/main/win_install.bat
```

```
win_install.bat
```

**Uninstall (Windows)**
Double-click `win_uninstall.bat` (or run from Command Prompt) to remove the `py11` environment, the source directory, and the launchers. Miniconda is left untouched.

```
win_uninstall.bat
```

### 4.3.2 Conda Environment

**I. Prepare the directory and clone the repository**
Establish a dedicated directory for the software and clone the source repository. To synchronize an existing installation with the latest updates, execute `git pull`.

```
mkdir -p ~/software/ascec04
```

```
git clone https://github.com/manuel2gl/qft-cosmic-ascec.git ~/software/ascec04/
```

**II. Install Miniconda (Skip if Conda is already installed)**
Download and run the installer. During installation, type `yes` when asked to initialize conda.

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

```
bash Miniconda3-latest-Linux-x86_64.sh
```

### III. Create and configure the environment

Create a clean Python 3.11 environment named `py11` (or your preferred name).

```
conda create -n py11 python=3.11 -y
```

### IV. Set up Aliases

To make running the programs easier, add shortcuts to your shell configuration. By pointing directly to the environment's Python binary, no `conda activate` is needed at runtime. First, find your conda base path:

```
conda info --base
```

Open your `.bashrc` file:

```
micro ~/.bashrc
```

Scroll to the bottom of the file and paste the following lines, replacing `<conda_base>` with the path from the previous command (e.g., `/home/user/miniconda3`):

```
# COSMIC ASCEC aliases
alias ascec='<conda_base>/envs/py11/bin/python $HOME/software/ascec04/ascec-v04.py'
alias cosmic='<base>/envs/py11/bin/python $HOME/software/ascec04/cosmic-v01.py'
```

*Save the file (Ctrl+S) and exit (Ctrl+Q). Then, reload the configuration:*

```
source ~/.bashrc
```

### V. Install Dependencies

Activate the environment and install the required libraries.

```
conda activate py11
```

```
conda install numpy scipy matplotlib scikit-learn -y
```

```
conda install -c conda-forge cclib openbabel -y
```

```
conda install -c conda-forge 'xtb>=6.7'
```

```
pip install orca-pi
```

*Note: You can install both parsers for maximum compatibility with different ORCA versions. OPI is installed via pip even in conda environments.*

### VI. Verify Installation

Check that the commands are working (no environment activation needed):

```
python -c "import numpy, scipy, matplotlib, sklearn, cclib, opi; print('All packages OK')"
```

```
ascec --version
```

```
ascec --help
```

```
cosmic --help
```

```
orca -v
```

```
obabel -V
```

```
(base) manuel@unix:~$ ascec -h

usage: ascec [OPTIONS] COMMAND [ARGUMENTS]

ASCEC - Annealing Simulado con Energia Cuantica
(Simulated Annealing with Quantum Energy)
Configurational sampling via Monte Carlo with quantum mechanical evaluation

positional arguments:
COMMAND        Input file or command (opt, ref, sort, cosmic, diagram, etc.)
ARG1           Command-specific argument (e.g., template file, mode)
ARG2           Additional command-specific argument

options:
-h, --help     show this help message and exit
-v             Increase verbosity level (use -v, -v2, -v3, etc.)
--standard     Use standard Metropolis criterion instead of modified
--nosum        Skip summary file generation during sort
--justsum      Generate summary file only without sorting structures
--nobox        Disable generation of box-visualization XYZ files
--maxprint     Keep all intermediate files from every stage (default: miniprint,
clean up at end)
-V, --version  Display version information and exit

COMMANDS:
Run an annealing job (the COMMAND is the .asc input file itself):
ascec input.asc                Run simulated annealing on input.asc
ascec input.asc rN             Build N replicate runs (e.g. r3 = 3 replicas)
```

# 5 Web Input Generator

ASCEC-v04 includes a browser based tool to generate input files automatically, fetch structures from PubChem, and visualize the simulation box in 3D.

## 5.1 Features

- **PubChem Integration:** Automatically converts SMILES, IUPAC names (e.g., "aspirin"), or brand names into 3D XYZ coordinates using the PUG REST API [8].

  Compounds search: https://pubchem.ncbi.nlm.nih.gov/compound/

- **3D Visualization:** Uses 3Dmol.js [9] to preview the simulation box, showing the boundaries and initial molecular placement.

- **Protocol Builder:** easy creation of multi-stage workflows (e.g., `opt`, `cosmic`, `ref`).

## 5.2 Accessing the Tool

You can access the generator in two ways:

1. **URL:** Visit https://manuel2gl.github.io/qft-cosmic-ascec/

2. **Direct:** Run the following command in your terminal:

```
ascec input
```

## 5.3 GUI Modes

The tool offers two complementary interfaces, selectable from the top navigation bar: **Smart Mode** (the default) and **Advanced Generation**. Both produce the same ASCEC input file; they differ only in how much of the configuration is exposed and how much is automated.

### Smart Mode

Smart Mode is a guided, automated workflow designed to get a valid simulation running with minimal input. The typical sequence is:

1. **Build the system.** Enter any PubChem identifier (name, SMILES, CAS, or InChI, e.g. `water`, `64-17-5`, `CCO`) in the *Molecule* → *XYZ Converter* to fetch 3D coordinates, and set the number of copies for each species.

2. **Set the simulation box.** Specify the box length directly, or enable *auto* to compute it from a target packing percentage. The molecules and box boundaries are previewed live in the 3D viewer.

3. **Choose the parameters.** A guided form selects sensible defaults for the annealing, pre-optimization, refinement, and COSMIC settings, so you only adjust the key choices.

4. **Review and download.** A pipeline summary shows every stage of the workflow before you download the generated input file.

## 5.4 Interface Overview

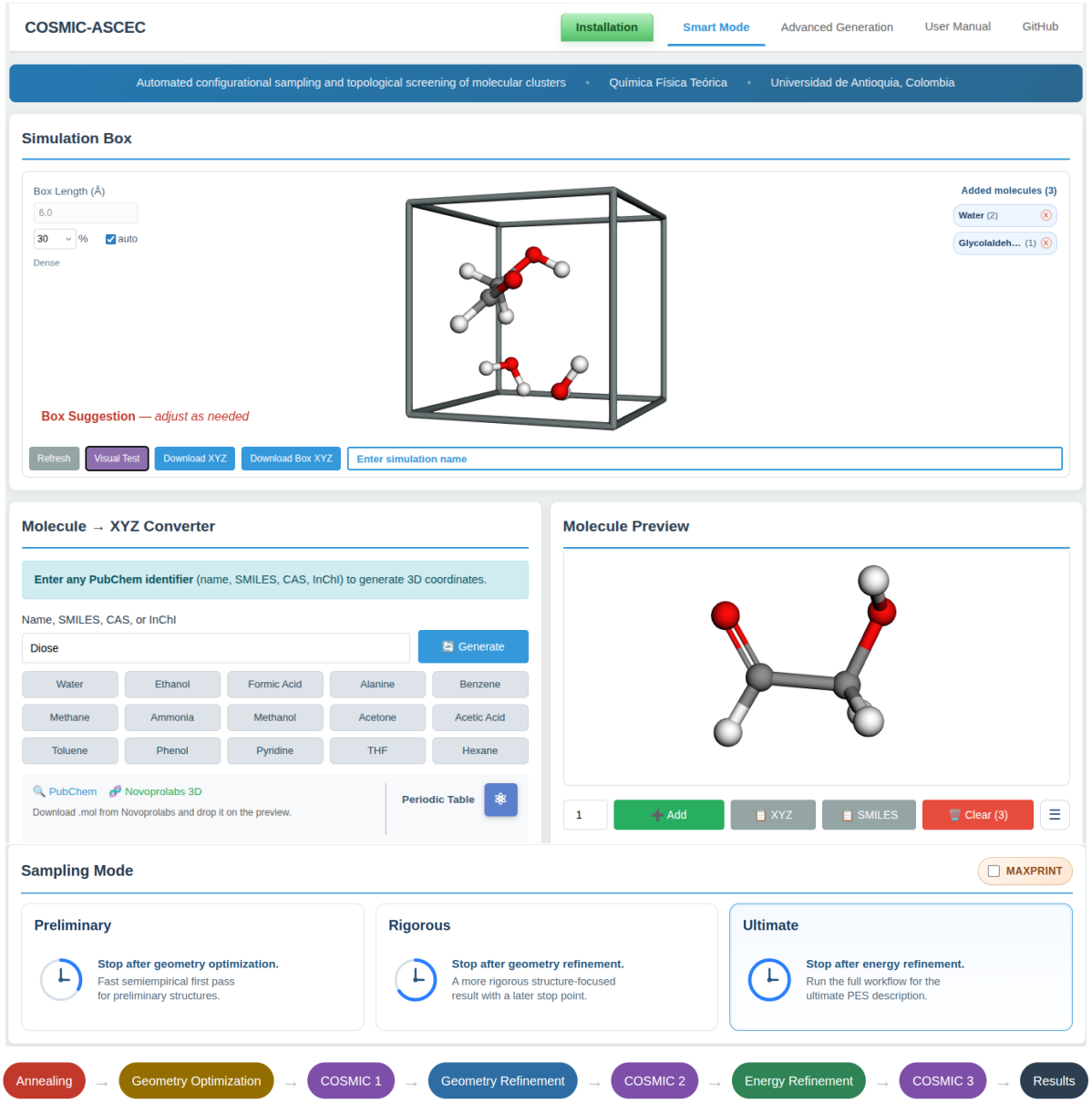



Figure 2: Web Input Generator interface with 3D visualization and parameter forms.

# 6 Quick Start: A Black-Box Example

## 6.1 Glycolaldehyde-Water

This section presents a minimal example that demonstrates the core functionality of COSMIC–ASCEC through the hydration of glycolaldehyde. It is intended as a black-box test that can be run without prior knowledge of the underlying theory or the input file syntax, providing a convenient way to verify the installation and obtain a first overview of the program's workflow and results.

The system consists of one glycolaldehyde and two water molecules. This is a particularly instructive test case because glycolaldehyde exhibits both **configurational** (intermolecular arrangement) and **conformational** (internal dihedral rotation) degrees of freedom, making it richer than a purely rigid-body system.

We begin by generating the initial coordinates with the web input generator. Glycolaldehyde is the simplest member of the diose family, so it can be located in the generator by searching for the name *diose* (Figure 3).

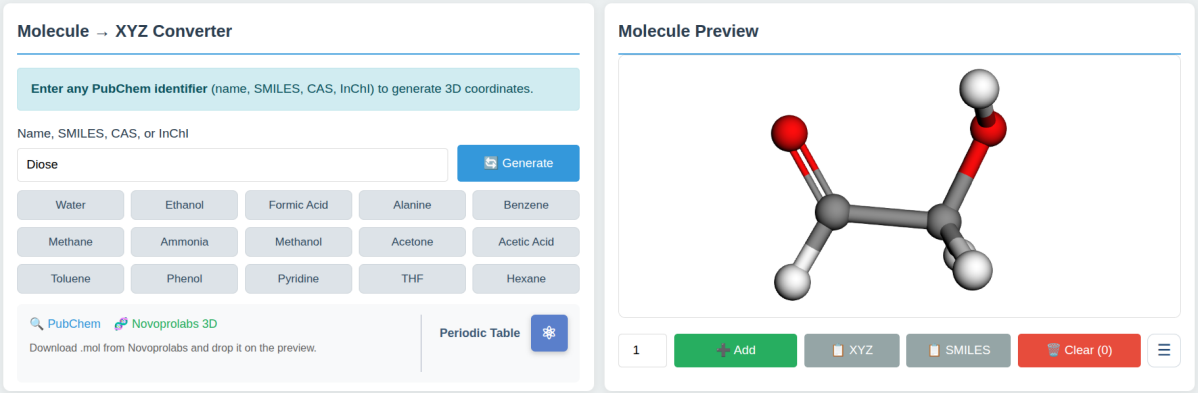



Figure 3: Generating initial coordinates for glycolaldehyde and water using the web interface.

A crucial step in the setup is superposition: both the glycolaldehyde and the water molecules are placed at the exact center of the simulation box, ensuring that the stochastic search starts from an unbiased configuration. The resulting setup should resemble Figure 4, so that the center of mass for each molecule occupies the same region of space before the simulation begins.

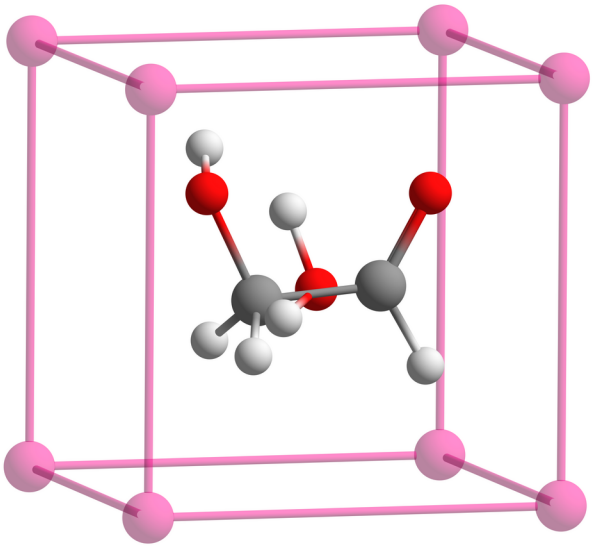


Figure 4: Initial simulation box containing glycolaldehyde and water superposed at the center.

In this example we select a preliminary run (Figure 5), which uses the GFN2-xTB method for both the single annealing stage and the subsequent candidate pre-optimization; motifs are then identified by the COSMIC clustering algorithm. This is an **optimization only** workflow (no frequency calculations), which keeps the process fast but does not guarantee that the resulting structures are true local minima.

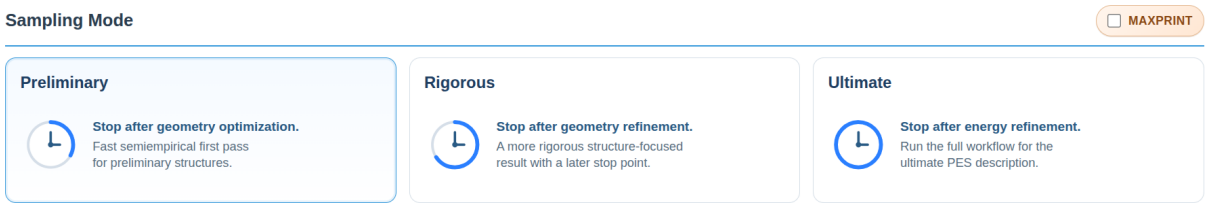



Figure 5: Preliminary sampling mode selection.

We enable the conformational option at the annealing theory level (Figure 6) and then download the `.asc` input file.

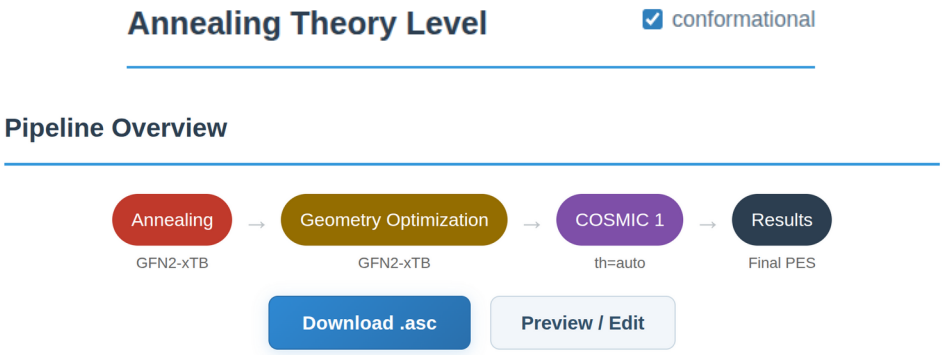



Figure 6: Conformational check and input download.

The necessary input file, `gaw.asc`, is provided in the examples folder.

```
/
└── examples/
    └── gaw.asc
```

To run the simulation, simply execute the program. This calculation is configured to use eight processor cores.

```
ascec gaw.asc
```

Upon execution, a progress bar is displayed.

```
=== COSMIC ASCEC ===
------------------------------------------------------------
Progress [.............................] 0.0%
------------------------------------------------------------
[1/3] Annealing ...
```

Here, `[1/3]` indicates that the program is in the first of three main stages: Annealing, Geometry Optimization, and COSMIC screening.

Once the run is complete, the terminal will display a summary message.

```
=== COSMIC ASCEC ===
------------------------------------------------------------
Progress [##############################] 100.0%
------------------------------------------------------------
[1/3] Annealing ✓
[2/3] geometry_optimization ✓
[3/3] cosmic (80→37) ✓

Workflow finished
Start: 2026-06-23 17:43:40
End:   2026-06-23 17:44:17
Total wall time: 36s 835ms
------------------------------------------------------------
Warning: No true minima can be assured without frequency calculations.
Protocol summary saved to protocol_summary.txt

✓ Miniprint: 23.8 MB (use --maxprint to keep all files)
```

In approximately 37 seconds, ASCEC autonomously executed a three-stage pipeline:

1. **Annealing** (66.3%): The stochastic search explored the configurational and conformational space of the glycolaldehyde–two-water complex using GFN2-xTB energies. Starting from a "Big Bang" superposition, it generated **80 accepted configurations** while cooling the system from 300 K down to near 0 K.

2. **Geometry Optimization** (33.7%): All 80 candidate structures were optimized at the GFN2-xTB level, relaxing each configuration into the nearest local energy basin and producing a set of pre-optimized structures.

3. **COSMIC** ($< 0.1\%$): The clustering algorithm analyzed the 80 optimized structures using a reduced physicochemical feature vector and identified **37 motifs**, i.e., topologically distinct families of hydrated glycolaldehyde.

A representative subset of these motifs is shown in Figure 7; each of them corresponds to a different interaction pattern between water and glycolaldehyde, with a certain degree of redundancy remaining.

Note the warning in the terminal output. Because this workflow uses **Opt** (optimization only, without frequency calculations), the COSMIC module operates with a **reduced feature vector**. This is sufficient for a rapid preliminary survey of the PES, comparable to the level of analysis performed by many conformational-sampling programs. However, without frequency analysis the structures cannot be rigorously confirmed as true local minima, since some may correspond to saddle points. For publication-quality results, a complete **Opt+Freq** workflow should be used (as demonstrated in the Water Hexamer example); this yields a fully characterized PES with verified minima and a Boltzmann-weighted population distribution. The trade-off between speed and rigor is ultimately the user's decision and depends on the goals of the study.

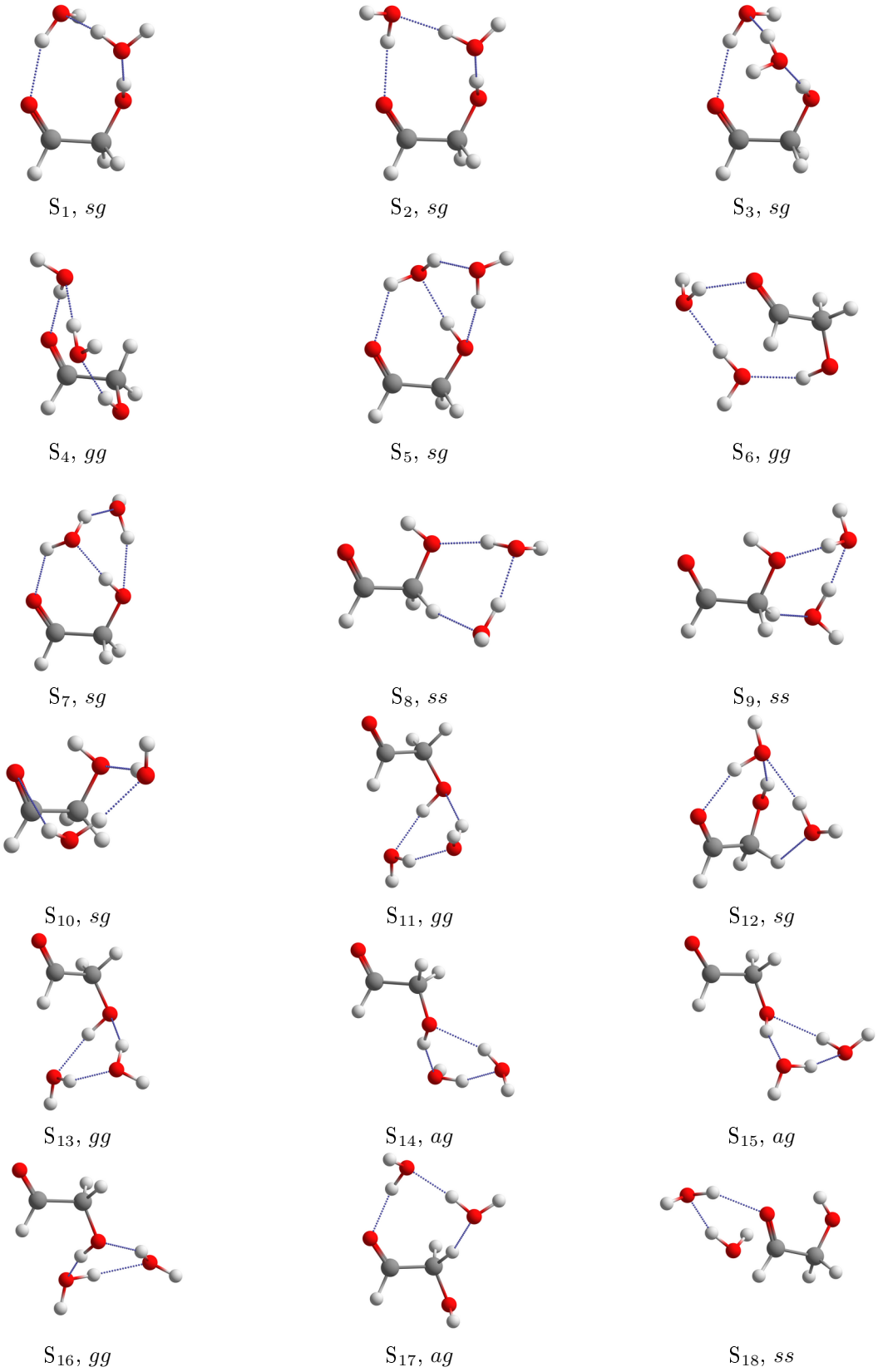



Figure 7: First 18 of the 37 preliminary glycolaldehyde–water motifs.

# 7 Water Hexamer

This section demonstrates the entire COSMIC–ASCEC workflow using the web generator to build the input file from scratch. Unlike the Quick Start example, this pipeline includes a geometry refinement stage, producing a fully characterized PES with verified minima and Boltzmann populations. We use the water hexamer $(H_2O)_6$ as a well studied benchmark system [10].

## 7.1 Molecular Coordinates Generation

Open the Web Input Generator at `https://manuel2gl.github.io/qft-cosmic-ascec/` (or run `ascec input` in the terminal). Using Smart Mode, search for "water" in PubChem to fetch its 3D structure, then add six copies to the simulation box. (Figure 8).

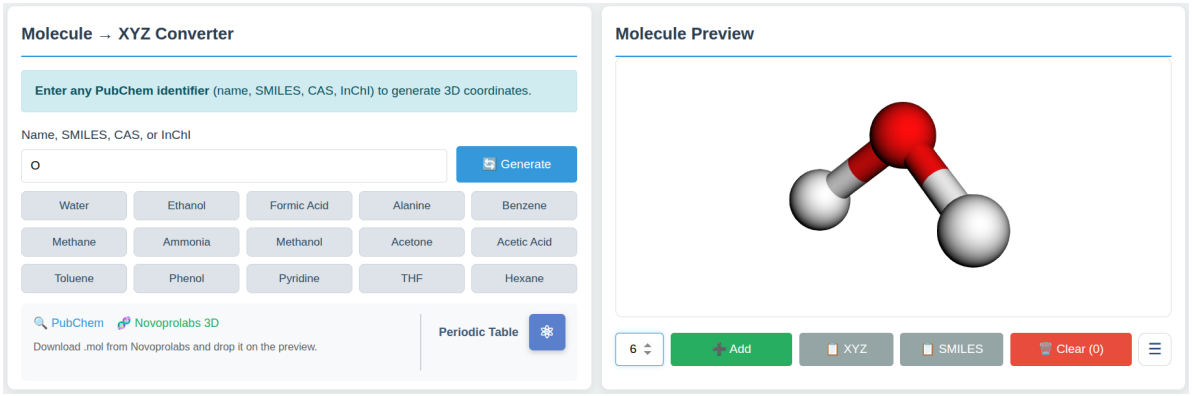



Figure 8: Generating six water molecules using the PubChem integration.

## 7.2 Mode Selection

The web generator provides several predefined modes. A **preliminary** run uses no replicas and a single low cost level of theory for a quick mapping of the PES. The **rigorous** and **ultimate** modes add a higher level refinement stage, producing a fully resolved PES with any level of theory available on the backend.

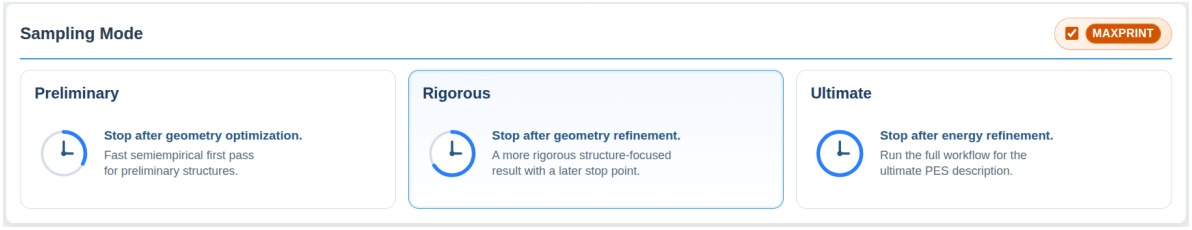



Figure 9: Rigorous mode selection.

As shown in Figure 9, the `maxprint` option is enabled by default in rigorous and ultimate modes: because these workflows involve expensive calculations, the user may prefer to retain all intermediate I/O files for further analysis. If it is disabled, `miniprint` is used instead and only the I/O files of the final ensemble are kept, saving disk space.

## 7.3 Simulation Box

The simulation box defines the cubic region where molecules are confined during the annealing search. The key parameter is the **effective packing percentage** $\phi$, which controls how full the box is: a lower value produces a more dilute (spacious) box, while a higher value makes it denser. The default is 30% for preliminary runs, and 20% for rigorous and ultimate modes, which provides additional free space for the molecules to reorganize during the more exhaustive search. The algorithm that converts $\phi$ into a box length, together with the underlying geometric reasoning, is described in detail in Section 10.

The web generator (and ASCEC itself) automatically estimates the appropriate box size based on the molecular volumes and, for systems with hydrogen bond donors and acceptors, the additional space required by the H-bond network. For the water hexamer at the 20% effective packing used in rigorous mode, the suggested box length is approximately 7.3 Å (Figure 10). A more detailed box analysis using the standalone `ascec box` command is presented in Section 11.

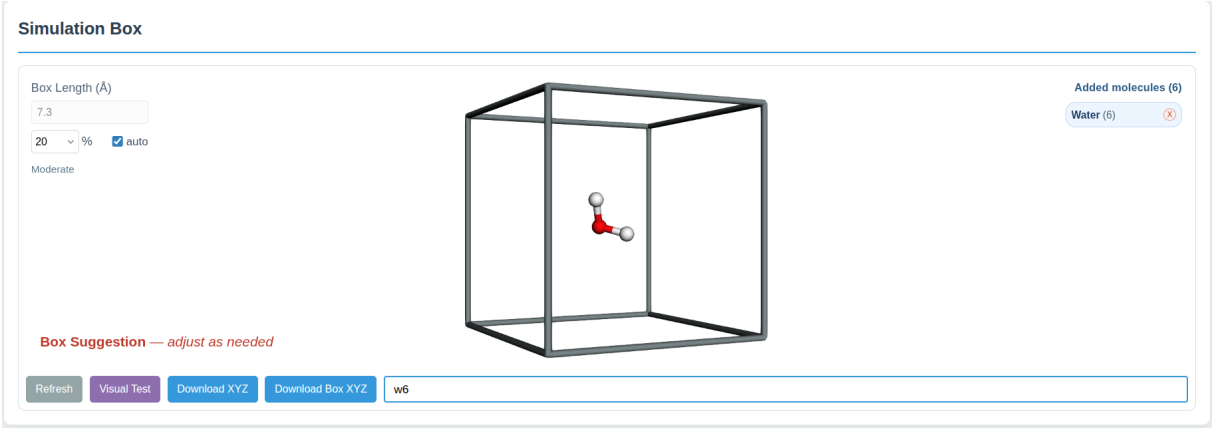



Figure 10: Simulation box setup for the water hexamer at 20% effective packing ($L \approx 7.3$ Å).

A visual test mimics the random rigid body moves performed by the ASCEC module, letting the user check whether the suggested box is adequate. This helps avoid both compressed boxes, which leave no room for cluster formation, and oversized boxes, in which the system cannot reliably evolve during the annealing procedure.

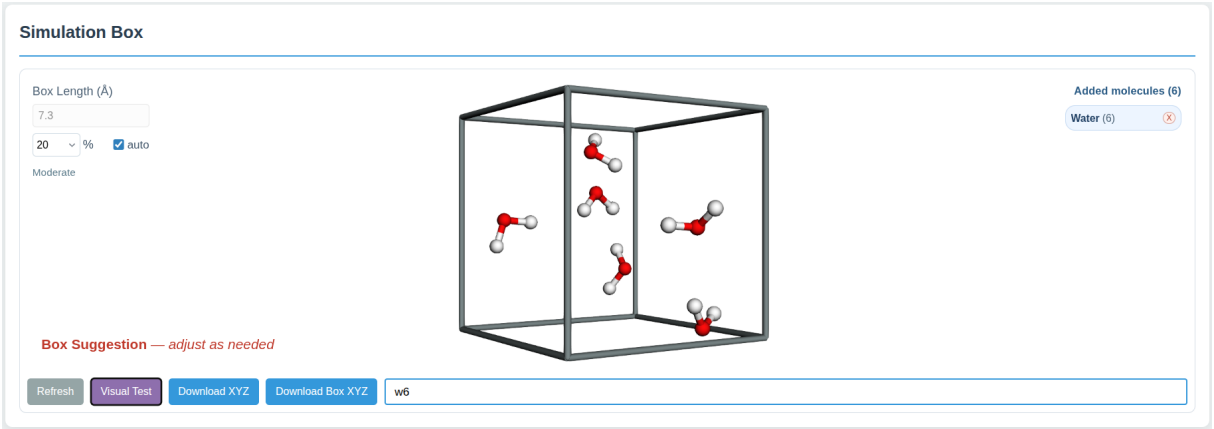



Figure 11: Water hexamer simulation box visual test

## 7.4 Annealing Parameters

Smart Mode provides a guided interface for selecting the annealing settings (Figure 12).

It supplies sensible defaults that the user can override as needed. The key parameters are:

- **Conformational:** Controls sampling of internal (torsional) degrees of freedom. For a *single* fragment it is enabled automatically with a sampling percentage of 100%. For *multiple* fragments it is optional (off by default); when enabled it defaults to 30%.
- **Annealing method:** GFN2-xTB semiempirical, for fast energy evaluations during the stochastic search.
- **nprocs:** Number of processors used for the QM calculations.
- **Concurrent & Replicas:** Three concurrent (triplicated) annealing runs to improve sampling coverage.
- **Charge & Multiplicity:** The system's overall charge and spin multiplicity; the GUI proposes a multiplicity, but it should always be reviewed.
- **Temperature schedule:** Geometric quenching, which gradually lowers the acceptance probability of high energy states.
- **Box:** With auto box enabled, the packing fraction can be adjusted; with it disabled, a custom box length can be entered.

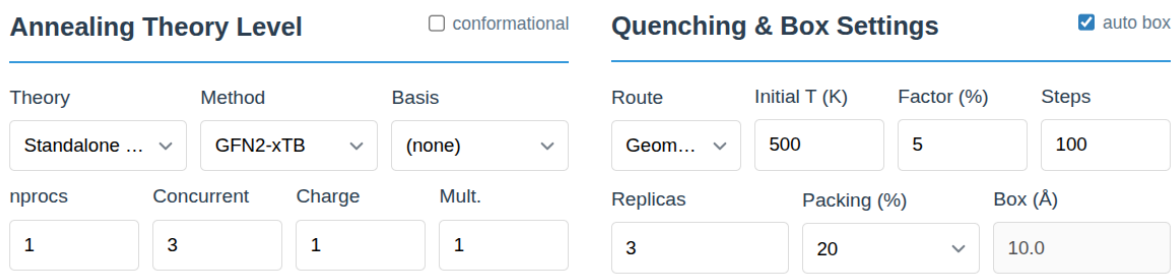



Figure 12: Annealing parameter selection.

## 7.5 Geometry optimization and refinement

A geometry optimization is first carried out with the GFN2-xTB semiempirical method to relax the accepted structures and enable a first COSMIC screening.

The refinement step re-optimizes the unique motifs identified by the first clustering pass at a higher level of theory (Figure 13). For this run we use `wB97X-D3BJ/def2-TZVPP` with a full **Opt+Freq** treatment, which ensures that the final structures and their relative energies are accurate enough for meaningful thermodynamic analysis. Crucially, the frequency calculation confirms each structure as a true local minimum.

A second COSMIC analysis is then applied to the refined ensemble to produce the final set of unique motifs and the Boltzmann population distribution.

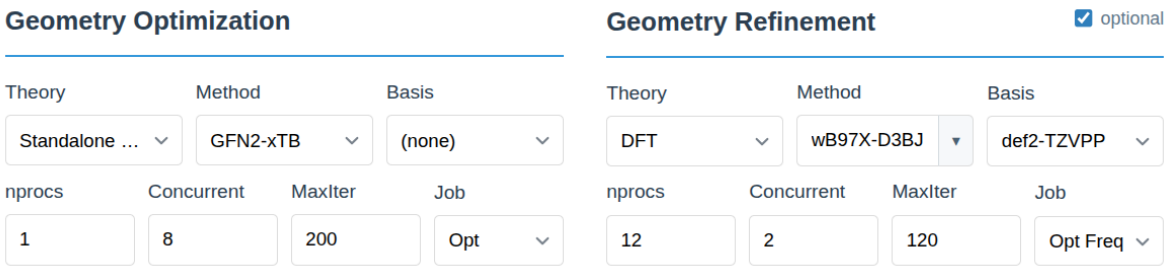



Figure 13: Geometry optimization & Refinement theory level.

## 7.6 Energy Refinement & COSMIC Screening

An optional energy refinement is reserved for the ultimate mode, which is why it is disabled in this rigorous run. The COSMIC screening parameters are also set here: in `automatic` mode the protocol determines the best clustering threshold during the first COSMIC pass (after the geometry-optimization stage; see Section 11.3.2 for details) and reuses that same threshold for the final COSMIC stage (Figure 14).

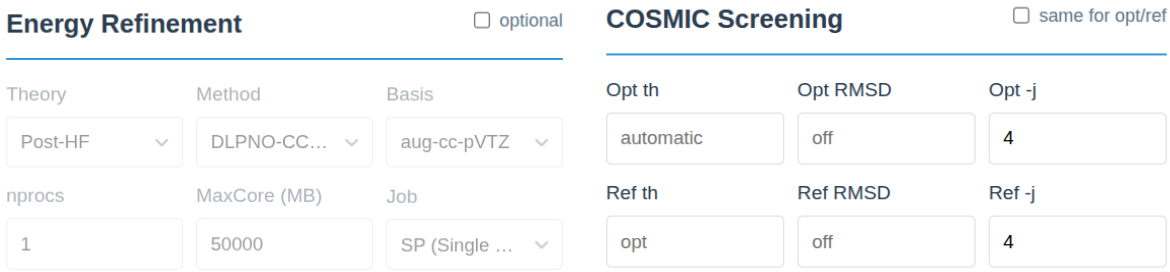



Figure 14: Energy Refinement & COSMIC Screening.

## 7.7 Pipeline Review and Execution

Before generating the input file, the interface shows the complete execution pipeline (Figure 15). This summary allows you to verify each stage before downloading the `.asc` file.

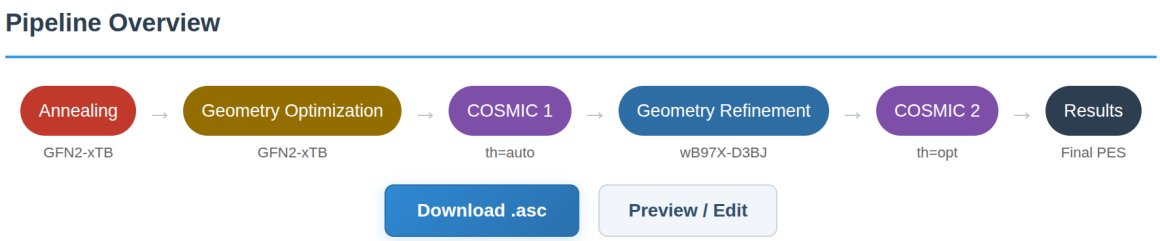



Figure 15: Complete pipeline review showing all workflow stages.

Download the generated `.asc` file and place it in your working directory, adjust the orca input lines as needed.

```
#orca input2
#name
#rescue(HF-3c/freq)
! wB97X-D3BJ def2-TZVPP def2/J RIJCOSX DEFGRID3
! Opt Freq TightOpt TightSCF
%pal
nprocs 12
end
%geom
maxiter 120
end
* xyz 0 1
#
*
```

Then execute the protocol:

```
ascec w6.asc
```

Because the full pipeline can run for hours or days, it is normally left running unattended. There are two ways to detach it from the terminal: launch it in the background with the shell ampersand, (`ascec w6.asc &`), or simply press `Ctrl+D` while the run is attached.

On `Ctrl+D` the program relaunches itself fully detached from the terminal session and prints the background PID; the job keeps running and the shell is freed. Closing the terminal does not stop a detached job.

The `ascec status` command opens an interactive viewer of all ASCEC jobs on the machine (a registry shared across terminals), so a detached run can be inspected at any time:

```
ascec status
```

It lists the running and recent jobs with their ID, PID, input file, and start time. From the prompt you can attach to a job and watch its live progress (`V <id>`), terminate it (`K <id>`), refresh the list (`R`), or quit the viewer (`Q`), quitting the viewer leaves the jobs running.

```
ASCEC STATUS  (2026-06-25 20:28:46)

Running:
ID      PID   INPUT FILE           STARTED
---  -------  ------------------  -------------------
1    1131777  w6.asc              2026-06-19 16:54:48

-----------------------------------------------------------------
[V <id>] Attach/View   [K <id>] Kill   [R] Refresh   [Q] Quit
-----------------------------------------------------------------
Choice: v1
```

```
=== COSMIC ASCEC ===
------------------------------------------------------------
Progress [####################.........] 68.6%
------------------------------------------------------------
[1/5] annealing ✓
[2/5] geometry_optimization ✓
[3/5] cosmic ✓ (313→49)
[4/5] geometry_refinement 21/49 ...

Attached: job 1 (at.asc)   updated: 2026-06-25 14:29:05
Ctrl+C or Ctrl+D to detach (job keeps running)
Input realpath: /home/protio/Documentos/at/at.asc
```

The automated protocol will sequentially run all stages: Annealing, Geometry Optimization, COSMIC, Geometry Refinement, and Final COSMIC. Depending on the theory levels and number of replicas, the computation may take several hours to a few days.

The run is resumable. Each stage checkpoints its progress to a cache file, so if the workflow is interrupted (or deliberately stopped), simply re-issuing the same command, `ascec w6.asc`, picks up where it left off: completed stages are read from the cache and skipped, and execution continues from the first unfinished stage.

## 7.8 Results

The automated protocol funnels the raw stochastic search down to a compact set of fully characterized minima. The three triplicated annealing runs produced **215 accepted configurations**, all of which were pre-optimized at the GFN2-xTB level. A first COSMIC pass reduced this ensemble to **39 motifs**, which were then refined at the `wB97X-D3BJ/def2-TZVPP` level with frequency analysis. A second COSMIC pass on the refined ensemble yielded **22 unique minima**. The complete pipeline ran in roughly **1 day and 3 hours** on a standard workstation, with the DFT refinement stage accounting for ∼99.8% of the wall time; the annealing and pre-optimization stages together took only a few minutes.

The final clustering applied a topological threshold of $\tau = 2.0$. Its built-in validation initially flagged 23.1% of the refined structures as critical (incomplete or non-minimum); these were automatically re-processed, and after the redo passes the critical fraction dropped to 0%, confirming that all 22 representatives correspond to genuine local minima. Figure 16 shows the threshold diagnostic that selects $\tau$ and the resulting dendrogram of the 22 clusters.

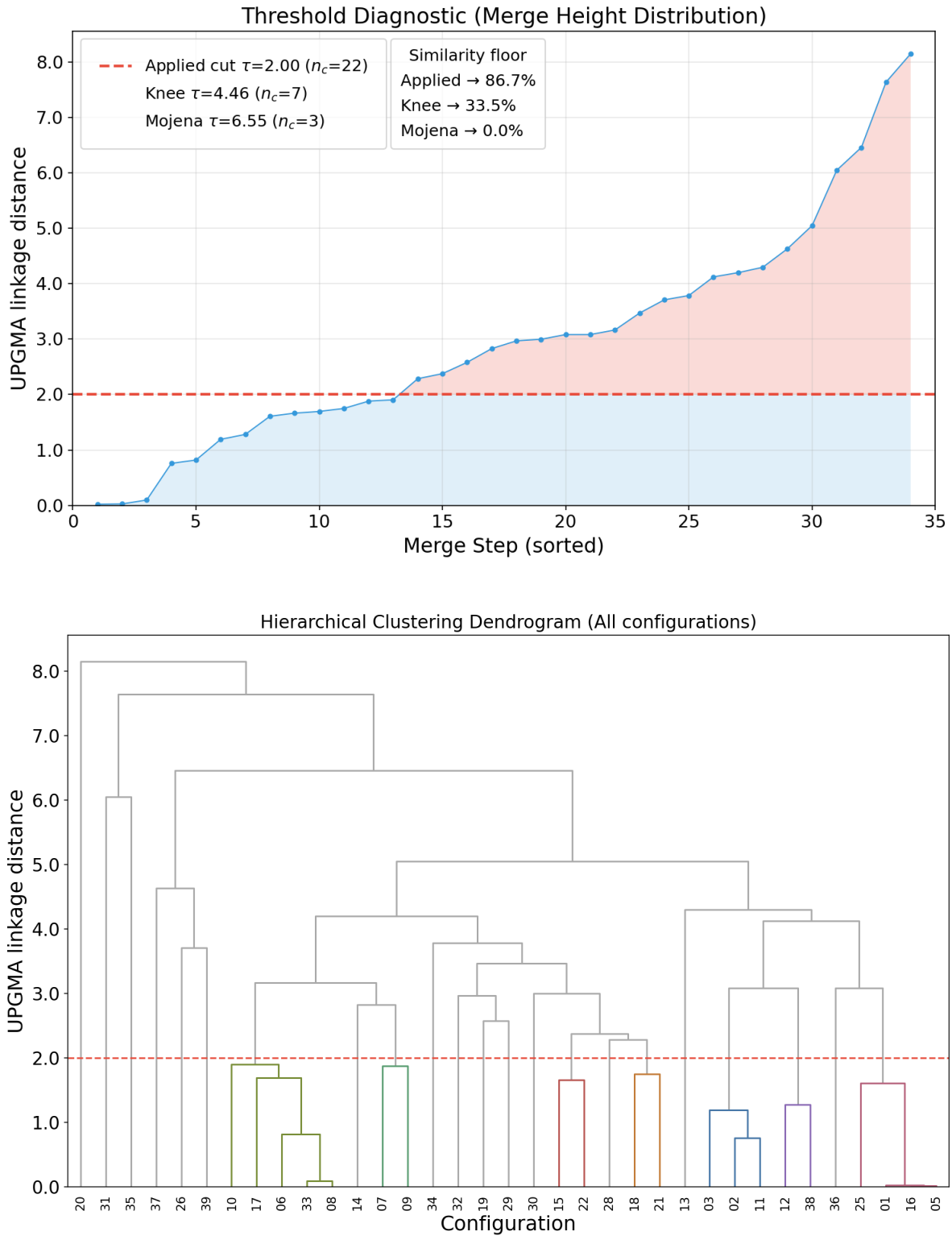
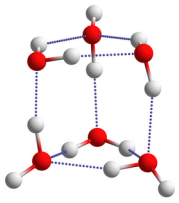
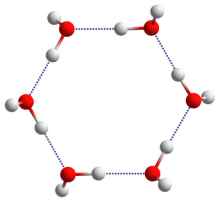


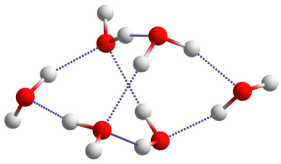

Figure 16: Final COSMIC analysis of the refined ensemble: the threshold diagnostic used to fix $\tau = 2.0$ and the dendrogram grouping the refined structures into 22 unique motifs.



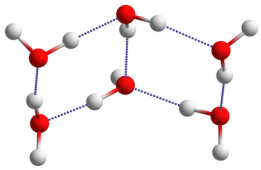
$S_1$, $\Delta\Delta G = 0.00$



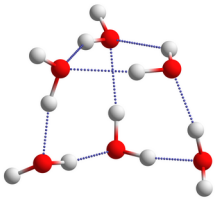
$S_2$, $\Delta\Delta G = 0.25$



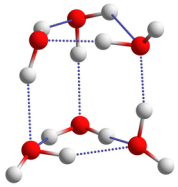
$S_3$, $\Delta\Delta G = 0.26$



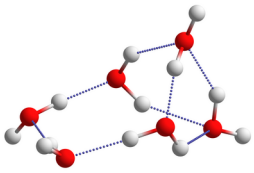
$S_4$, $\Delta\Delta G = 0.32$



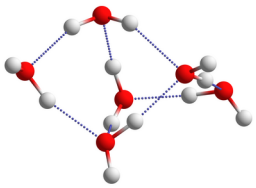
$S_5$, $\Delta\Delta G = 0.56$



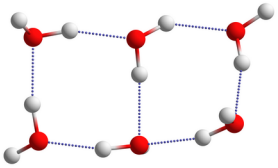
$S_6$, $\Delta\Delta G = 1.20$



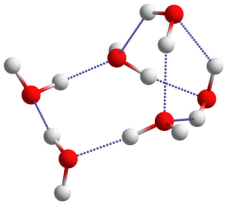
$S_7$, $\Delta\Delta G = 1.46$



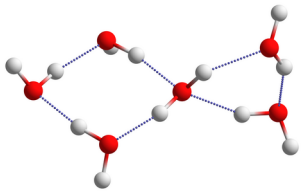
$S_8$, $\Delta\Delta G = 1.66$



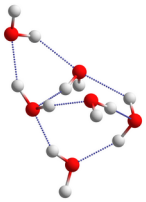
$S_9$, $\Delta\Delta G = 1.70$



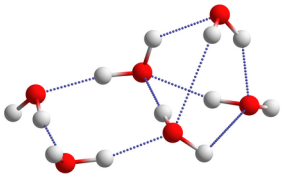
$S_{10}$, $\Delta\Delta G = 1.89$



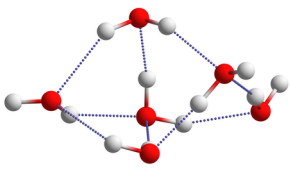
$S_{11}$, $\Delta\Delta G = 2.48$



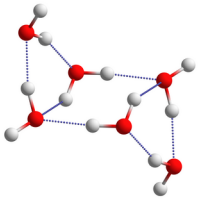
$S_{12}$, $\Delta\Delta G = 2.66$



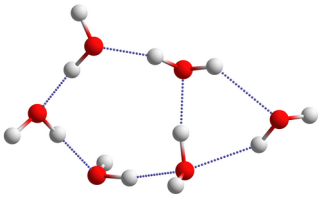
$S_{13}$, $\Delta\Delta G = 2.79$



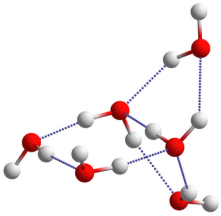
$S_{14}$, $\Delta\Delta G = 2.85$



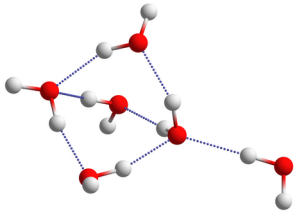
$S_{15}$, $\Delta\Delta G = 3.07$



$S_{16}$, $\Delta\Delta G = 3.27$



$S_{17}$, $\Delta\Delta G = 3.31$



$S_{18}$, $\Delta\Delta G = 3.69$

Figure 17: First 18 of the 22 refined water hexamer minima, labeled with their relative Gibbs free energy $\Delta\Delta G$ (kcal/mol) at 298.15 K.

The Boltzmann population (298.15 K) is dominated by a few low lying minima: the five most stable motifs hold ∼**88%** of the population, all within ∼0.6 kcal/mol of the global minimum. A subset is shown in Figure 17.

The final ensemble is only as reliable as the refinement method: faster methods (semiempirical, GFN2-xTB) give a cheap preliminary survey, but accuracy depends on the high-level stage. Against reported DFT and MP2 data for small water clusters [10], the ASCEC protocol recovers the known dominant motifs at a reasonable cost on a standard workstation.

**Computational efficiency.** The hierarchical design reserves expensive quantum-chemical work for the few structures that matter (Figure 18). The ASCEC search and xTB pre-optimization explore the space and generate hundreds of candidates for a small fraction of the runtime, the water hexamer's preliminary survey finishes in ∼3 minutes, while rigorous DFT geometry optimization and, in the ultimate protocol, DLPNO-CCSD(T) energy refinement dominate ($\gtrsim 99\%$) the wall time.

Since COSMIC collapses the ensemble to a handful of motifs *before* these steps run, costly calculations are minimized while the dominant minima are retained, letting the user trade accuracy against cost (preliminary, rigorous, ultimate) predictably.

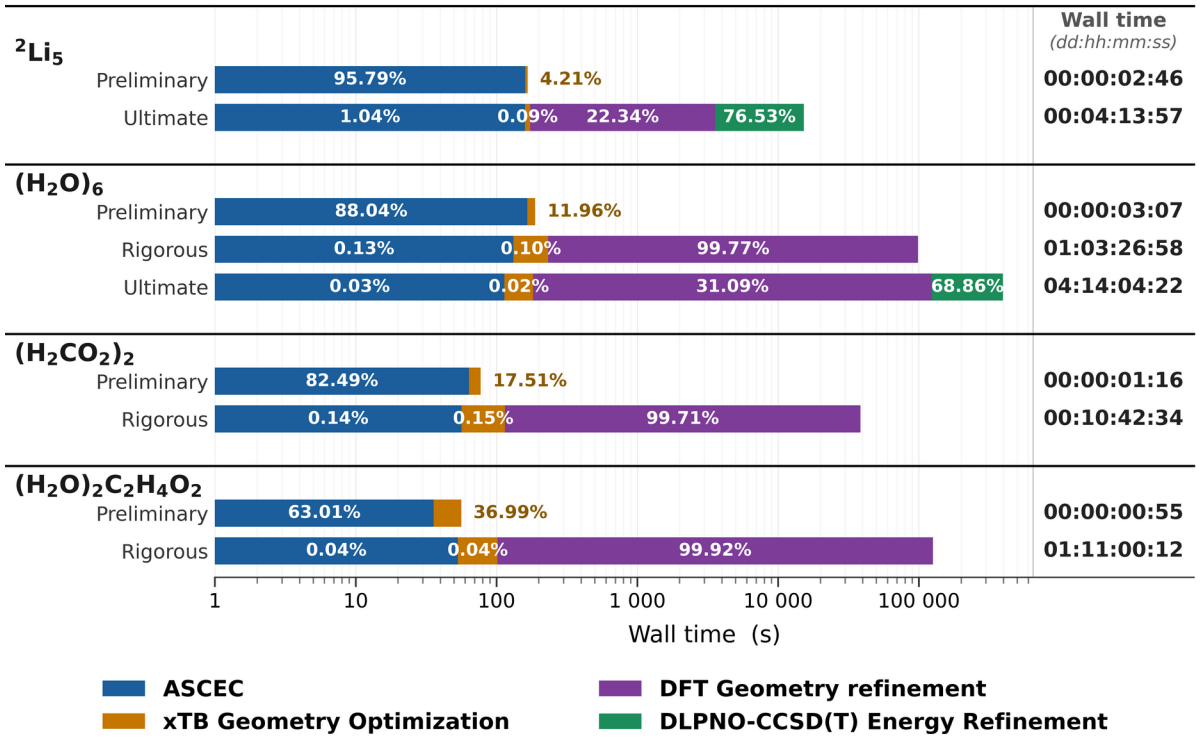



Figure 18: Wall time breakdown of the COSMIC–ASCEC pipeline stages for four benchmark systems, including the water hexamer $(H_2O)_6$. Total wall time spent in each stage shown in logarithmic time axis.

# 8 Advanced Usage & Reference

# 9 Input File Format

The ASCEC input file (typically saved with an `.asc` extension) is a comprehensive "master" document. To make it easy to read and edit, it is logically divided into three main blocks: the **Simulation Parameters**, the **Execution Protocol**, and the **Embedded QM Templates**.

*Note: The easiest way to generate this file without worrying about formatting is by using the ASCEC Web Input Generator.*

## 9.1 Part 1: Simulation Parameters & Coordinates

The first block of the input file relies on strict positional formatting. It defines the physical conditions of the annealing simulation and the initial coordinates of the molecules.

```
# Input for water pentamer Simulation

1 10            # Line 1: Simulation Mode & Number of Config
                # 1: Annealing; 0: Random configurations
5               # Line 2: Simulation Cube Length (Angstroms)
2               # Line 3: Annealing Quenching route
                # (1: Linear, 2: Geometrical).

100.0  10.0  50   # Line 4: Linear Quenching Parameters
500.0   5.0  100  # Line 5: Geometric Quenching Parameters

3000  50        # Line 6: Maximum MC Cycles per T and floor value

1.0 1.0         # Line 7: Maximum Displacement (A)
                # & Maximum Rotation (radians)
0 60            # Line 8: Conformational sampling (%)
                # & Maximum dihedral rotation (degrees)
2 orca            # Line 9: QM Program Index & alias
                # (2: ORCA, 3: for other, etc.)
b3lyp 6-31G*    # Line 10: Hamiltonian & Basis Set
                # (e.g., gfn2, b3lyp)(e.g., 6-31G*, def2-SVP)
1 8             # Line 11: nprocs (QM and ASCEC evaluations)
0 1             # Line 12: Charge & Spin Multiplicity

5               # Line 13: Number of Molecules (nmo)

# Lines below: Molecule Definition
# (Num Atoms, Label, followed by xyz coordinates, separated by *)

*
3
water1
O   0.000000  0.000000  0.000000
H   0.757000  0.586000  0.000000
H  -0.757000  0.586000  0.000000
*
3
water2
O   0.000000  0.000000  0.000000
H   0.757000  0.586000  0.000000
H  -0.757000  0.586000  0.000000
```

```
*   ... (W3-5)
```

Table 2: Detailed description of ASCEC input parameters (Lines 1-13).

| Line | Parameters | Description |
|---|---|---|
| 1 | [mode] [nconfigs] | **1**: Annealing, **0**: Random Generation. `nconfigs` is only used if mode is 0. |
| 2 | box_length | Length of the cubic simulation box edge in Å. |
| 3 | route | Temperature schedule: **1** (Linear) or **2** (Geometric). |
| 4 | [Ti][dT] [steps] | Linear: Start Temp (K), Step size (K), Number of steps. |
| 5 | [Ti] [fac] [steps] | Geometric: Start Temp (K), Reduction factor (%), Steps. |
| 6 | [max][floor] | Maximum Monte Carlo cycles per temperature step and floor value as it decreases (0.1) with each step. |
| 7 | [disp] [rot] | Max translation distance (Å) and max rotation (radians). Maximum displacement for each mass center, and maximum rotation of the principal axes of each molecule. The second field must be 0 for atomic clusters. |
| 8 | [prob][dih] | Probability of conformational move (%) and max dihedral change (°). |
| 9 | [code] [alias] | QM Program ID (**2**=ORCA, **3**=other) and the command alias to run it. |
| 10 | [method][basis] | Hamiltonian/Method (e.g., `gfn2`, `b3lyp`) and basis set (e.g., `def2-SVP`). For semiempirical methods the basis field is ignored; consult the ORCA manual for the correct syntax. |
| 11 | [cpu_Q] [cpu_A] | Processors passed to the QM program, and (optional) processors for ASCEC logic and parsing. |
| 12 | [chg][mult] | Total system Charge and Spin Multiplicity. |
| 13 | nmol | Number of molecules defined. For atomic clusters each atom must be considered a molecule. If atoms interacting with molecules are considered, then each atom not being part of a molecule must be considered a molecule. |

## 9.2 Part 2: The Automated Protocol & Embedded Templates

Following the molecular coordinates, the file contains the `# Protocol` block and the `Embedded Inputs`. The **Protocol** defines the automated execution pipeline as a series of sequential, comma-separated steps. It dictates the exact trajectory of your workflow: starting with stochastic annealing, moving to a preliminary pre-optimization and screening, and concluding with high-level quantum refinement and a final topological clustering to generate the Boltzmann ensemble.

```
# Protocol
# (annealing, (pre)optimization, clustering, separated by "," or ".")

.asc,
r2 --box10,
opt input1,
cosmic --th=2 -j4,
ref input2,
cosmic --th=2 -j4
```

```
Workflow:   Annealing → Opt → COSMIC → Ref → COSMIC_2
```

For maximum convenience, the required ORCA input templates that the protocol calls upon (e.g., `input1` and `input2`) are simply **embedded directly at the bottom** of the same file.

```
#orca input1
#name
#rescue(Native-GFN2-xTB/num)
! Native-GFN2-xTB
! Opt NumFreq
%pal
nprocs 4
end
%geom
maxiter 50
end
* xyz 0 1
#
*

#orca input2
#name
#rescue(HF-3c/freq)
! BP86 def2-SVP
! Opt Freq
%pal
nprocs 4
end
%geom
maxiter 10
end
* xyz 0 1
#
*
```

### Understanding the Execution Pipeline

Upon parsing the `.asc` file and its embedded templates, ASCEC autonomously executes the workflow sequentially. It begins with stochastic **Annealing** (`r2 -box10`) to generate raw candidate structures, immediately followed by pre-**Optimization** (`opt input1`) of the entire ensemble using the `Native-GFN2-xTB` template.

A preliminary **Screening** (`cosmic -th=2`) then applies topological clustering to filter out redundancies. The resulting motifs undergo high-level **Refinement** (`ref input2`) via the `BP86/def2-SVP` template. Finally, a **Final Screen** (`cosmic -th=2`) classifies the refined structures to yield the true unique minima and their Boltzmann population distribution.

***Note**: From this point forward, the advanced sections of this manual will teach you how to deeply understand each of these individual stages, master technical details, utilize custom external launchers, and execute these functions as standalone commands.*

# 10 Box Size Suggestion

ASCEC performs simulated annealing inside a hard-walled cubic box of side length $L$. Molecular (or atomic) centres of mass are confined within the walls, and trial moves that place a centre outside the box are rejected. Choosing $L$ correctly is critical: a box that is **too small** compresses the molecules, prevents free rotation, and introduces artificial steric clashes that distort the energy landscape; a box that is **too large** wastes annealing time sampling empty space and converges slowly.

The whole procedure is governed by a single, intuitive parameter: the **packing fraction** $\phi$ (given as a percentage, e.g. `-box20` for $\phi = 20\%$), which answers the question *"how full do you want the box to be?"* Two methods are used to compute an **effective volume** $V_{\text{eff}}$, but both feed the same box-length formula:

**Box length formula (both methods)**

$$L = \left( \frac{V_{\text{eff}}}{\phi} \right)^{1/3} \tag{6}$$

| Method | When used | $V_{\text{eff}}$ | What it captures |
|---|---|---|---|
| A | System has primary H-bonds | $V_{\text{mol}} + V_{\text{HB}}$ | Hull volume + H-bond ghost cylinders |
| B | No primary H-bonds | $L^3_{\text{diag}}$ | Diagonal-derived volume from molecular extents |

Table 3: The two routes to the effective volume. The selection is automatic.

Method A is preferred whenever the system can form hydrogen bonds, because it captures the directional, rigid geometry of H-bonded networks. For systems with no primary H-bonds (van der Waals clusters, noble-gas clusters, metal clusters) Method B estimates the required volume from the molecular extents. Both are fully general: they work for any number of molecules and for atomic, molecular, or mixed systems, requiring only the 3D coordinates and atomic numbers.

## 10.1 Molecular Volume via Convex Hull

Each atom is treated as a hard sphere whose radius is its **single-bond covalent radius** (Cordero *et al.*, 2008). The spheres of one molecule are assembled in their correct 3D geometry, and the **convex hull** of that surface — the smallest convex polyhedron enclosing it, as if a sheet were stretched tightly around the cluster — defines the molecular volume $V_{\text{mol}}$. No shape assumption is made: a single atom yields a sphere, a planar molecule a flattened polyhedron, a globular cluster a roughly spherical one.

Covalent radii (not van der Waals radii) are used because they define the irreducible hard core that other molecules cannot overlap. The softer van der Waals contact region is part of the *free space* between molecules, which is precisely what the packing fraction $\phi$ controls; using van der Waals radii would double-count that space and inflate the box.

| Element | H | C | N | O | F | S | Cl | Ar | Au |
|---|---|---|---|---|---|---|---|---|---|
| $r_{\text{cov}}$ / Å | 0.31 | 0.73 | 0.71 | 0.66 | 0.57 | 1.05 | 1.02 | 1.06 | 1.36 |

Table 4: Representative single-bond covalent radii used as sphere radii.

**Surface points: the Fibonacci sphere**

For a molecule with $N$ atoms at positions $\mathbf{r}_k$ and covalent radii $R_k$, a set of points approximating the outer envelope of each atomic sphere is generated with the **Fibonacci sphere** algorithm, which places $N_s = 50$ nearly uniformly distributed points per atom. For $i = 0, 1, \ldots, N_s - 1$:

$$\theta_i = \arccos\left(1 - \frac{2(i + 0.5)}{N_s}\right), \tag{7}$$

$$\varphi_i = \frac{2\pi\, i}{\Phi}, \qquad \Phi = \frac{1 + \sqrt{5}}{2} \approx 1.618 \ \text{(golden ratio)}, \tag{8}$$

$$\hat{\mathbf{d}}_i = \big(\sin\theta_i \cos\varphi_i,\ \sin\theta_i \sin\varphi_i,\ \cos\theta_i\big), \tag{9}$$

and the surface point for atom $k$ in direction $i$ is $\mathbf{p}_{k,i} = \mathbf{r}_k + R_k\, \hat{\mathbf{d}}_i$, giving $50\,N$ points in total. Unlike a simple $\{-1, 0, +1\}^3$ lattice of directions, the Fibonacci distribution contains **no coplanar clusters** of points. This matters because two different convex-hull libraries (e.g. Qhull in Python versus the incremental hull used by the web generator) can triangulate coplanar regions differently and report slightly different volumes; the Fibonacci construction removes that ambiguity and is deterministic to IEEE 754 double precision, so both implementations agree.

**Hull volume**

The hull volume is obtained from the **signed-tetrahedron formula**. For a hull with $n_F$ triangular faces, each with vertices $\mathbf{v}_1^{(f)}, \mathbf{v}_2^{(f)}, \mathbf{v}_3^{(f)}$:

$$V_{\text{mol}} = \left| \frac{1}{6} \sum_{f=1}^{n_F} \mathbf{v}_1^{(f)} \cdot \big(\mathbf{v}_2^{(f)} \times \mathbf{v}_3^{(f)}\big) \right|. \tag{10}$$

This is exact for any closed polyhedron; the only approximation is the discretisation of each atomic sphere into 50 points. For a single argon atom ($R = 1.06$ Å) the hull of 50 Fibonacci points gives $V_{\text{mol}} \approx 4.82$ Å$^3$, within 3.5% of the analytical sphere volume 4.99 Å$^3$ — a slight, conservative underestimate that is absorbed by the free-space fraction $(1 - \phi)$. If a hull library is unavailable, a bounding-box fallback $V_{\text{mol}} \approx 0.52\, V_{\text{bbox}}$ is used, the factor 0.52 approximating the typical hull-to-box ratio for organic molecules.

## 10.2 Molecular Extent

The **molecular extent** $E$ is the largest distance spanned by the molecule, including the covalent radii of the two terminal atoms:

$$E = \max_{j<k} \big(\|\mathbf{r}_j - \mathbf{r}_k\| + R_j + R_k\big). \tag{11}$$

It is the effective "diameter" of the molecule — the largest dimension it can occupy regardless of orientation. For a single atom, $E = 2R$.

## 10.3 Hydrogen-Bond Network Volume (Method A)

Hydrogen bonds are directional and rigid, operating at $\approx 2.5$ Å and forcing molecules into open arrangements with large voids (the same effect that makes ice less dense than liquid water). The hull volume measures only the molecular skeleton, so Method A adds a cylindrical "ghost volume" per potential H-bond to signal that the system needs extra room.

A system is treated as hydrogen-bonded when, across all molecules, there is at least one **donor** and one **acceptor**. Every H atom counts as one donor; N, O and F each count as one acceptor,

while S counts as 0.5 and Cl as 0.3 (their weaker, longer, less directional bonds). For each molecule the number of potential bonds is $n_{\mathrm{HB}} = \min(n_{\mathrm{donors}}, n_{\mathrm{acceptors}})$, and each is modelled as a cylinder of length $\ell_{\mathrm{HB}} = 2.5$ Å and radius $r_{\mathrm{HB}} = 1.2$ Å:

$$V_{\mathrm{HB}}^{(\mathrm{bond})} = \pi\, r_{\mathrm{HB}}^2\, \ell_{\mathrm{HB}} = \pi\,(1.2)^2(2.5) \approx 11.31\ \text{Å}^3, \qquad V_{\mathrm{HB}}^{(\mathrm{mol})} = n_{\mathrm{HB}}\, V_{\mathrm{HB}}^{(\mathrm{bond})}. \tag{12}$$

## 10.4 Effective Volume and Box Length

Summing over all $M$ molecules (copies included) gives the totals $V_{\mathrm{mol}}^{\mathrm{total}} = \sum_i V_{\mathrm{mol},i}$, $V_{\mathrm{HB}}^{\mathrm{total}} = \sum_i V_{\mathrm{HB},i}^{(\mathrm{mol})}$, and $\sum_i E_i$. The effective volume is then selected automatically:

$$\text{Method A (H-bonded):} \quad V_{\mathrm{eff}}^{(A)} = V_{\mathrm{mol}}^{\mathrm{total}} + V_{\mathrm{HB}}^{\mathrm{total}}, \tag{13}$$

$$\text{Method B (no H-bonds):} \quad V_{\mathrm{eff}}^{(B)} = L_{\mathrm{diag}}^3, \qquad L_{\mathrm{diag}} = \frac{1.5 \sum_i E_i}{\sqrt{3}}. \tag{14}$$

In Method B the numerator $1.5\sum_i E_i$ is the body diagonal of a cube in which all molecules, lined up end to end, would fit with a 50% buffer; dividing by $\sqrt{3}$ converts that diagonal to a cube edge, giving a "natural geometric size" for the system. The box length then follows from Eq. 6; for Method B it simplifies to $L^{(B)} = L_{\mathrm{diag}}/\phi^{1/3}$, so that $\phi = 1$ recovers $L = L_{\mathrm{diag}}$ and smaller $\phi$ grows the box to add free space.

### Why the cubic root matters

Because $L = (V_{\mathrm{eff}}/\phi)^{1/3}$, doubling the number of molecules doubles $V_{\mathrm{eff}}$ but increases $L$ by only $2^{1/3} \approx 1.26$, which is correct since the box volume scales as $L^3$. A naive linear rule $L \propto \sum_i E_i$ would instead double $L$ and produce a box $2^3 = 8$ times too large; for 100 methane molecules it would suggest $L \approx 277$ Å versus the $\approx 15$ Å given by the packing-fraction method.

## 10.5 Choosing the Packing Fraction

The packing fraction answers *"what fraction of the box is effectively occupied?"* — molecular hulls plus H-bond cylinders in Method A, or the diagonal-derived reference volume in Method B. The same `-box20` therefore means the same thing for water, methane, argon, or a mixed system; the algorithm computes $V_{\mathrm{eff}}$ and picks the method without user intervention. The web generator's preset modes set $\phi$ for you (30% for preliminary, 20% for rigorous and ultimate); when driving the command line directly, the following ranges are useful guidance:

| $\phi$ | Density regime | Typical application |
|---|---|---|
| 5%–10% | Very sparse | Initial conformational search |
| 10%–20% | Sparse | Most molecular cluster studies |
| 20%–30% | Moderate | Network or aggregate formation |
| 30%–50% | Dense | Condensed-phase packing, crystal seeds |
| 50%–65% | Very dense | Liquid-like; approaching the sphere-packing limit |

Table 5: Recommended packing fractions by density regime.

## 10.6 Worked Example: Water Hexamer

The water hexamer $(H_2O)_6$ illustrates Method A and reproduces the suggestions printed by `ascec box` (Section 11). Each water contributes a hull volume of $\approx 1.08$ Å$^3$ and, with two

donors and one acceptor, exactly one ghost cylinder ($n_{\mathrm{HB}} = 1$, $V_{\mathrm{HB}}^{(\mathrm{mol})} = 11.31$ Å$^3$). Summing over the six molecules:

$$V_{\mathrm{mol}}^{\mathrm{total}} = 6 \times 1.08 \approx 6.46 \text{ Å}^3, \qquad V_{\mathrm{HB}}^{\mathrm{total}} = 6 \times 11.31 = 67.86 \text{ Å}^3,$$
$$V_{\mathrm{eff}}^{(A)} = 6.46 + 67.86 = 74.31 \text{ Å}^3.$$

The H-bond ghost volume dominates ($\approx 10\times$ the hull volume), consistent with the open structure of water networks. Applying Eq. 6 reproduces the recommendations reported by the program:

| $\phi$ | $L$ / Å | Note |
|---|---|---|
| 5% | 11.4 | isolated clusters |
| 10% | 9.1 | cluster formation |
| 15% | 7.9 | network studies |
| 20% | 7.2 | **rigorous-mode default** |

Table 6: Box length for $(H_2O)_6$ from $V_{\mathrm{eff}} = 74.31$ Å$^3$ at several packing fractions.

The 20% value ($L \approx 7.2$ Å, rounded to 7.3 Å in the input file) is the box used for the rigorous water-hexamer run presented earlier in this manual, comfortably accommodating a six-membered ring or cage with O$\cdots$O $\approx 2.8$ Å. Both the Python (`ascec-v04.py`, Qhull) and JavaScript (web generator) implementations share the same covalent radii, Fibonacci formula, H-bond logic, cylinder parameters, and packing formula, so their suggested box lengths agree to the reported single decimal place.

# 11 Standalone Functions

**Commands for individual steps using ORCA 6.1.1.**

We need the input file (.asc) and the input QM file for pre-optimization with its respective launcher (see the examples folder for reference).

## 11.1 Annealing

### 11.1.1 Box Size

**Triplicated run for water hexamer**
Once the input file is generated, we can see the box size suggestions

```
/
└── w6.asc
```

```
ascec w6.asc box
```

```
(base) :~/.../examples/water_hexamer/orca6$ ascec w6.asc box

================================================================================
Box length analysis
================================================================================
```

```
1. Molecular volume and hydrogen bonding analysis:
--------------------------------------------------
Number of molecules to place: 6
Total molecular volume: 6.46 Å³
Total H-bond network volume: 67.86 Å³
Total effective volume: 74.31 Å³

3. Current box analysis:
-------------------------
Cube's length = 5.00 Å
Current effective packing: 59.5%
+ Molecular: 5.2%, H-bond network: 54.3%
Current free volume: 51 Å³ (40.5%)
Largest molecular extent: 1.13 Å
Extremely dense - may prevent proper H-bond formation

4. Recommendations for H-bonded systems:
----------------------------------------
• For isolated clusters: 11.4 Å (5% effective packing)
• For cluster formation: 9.1 Å (10% effective packing)
• For network studies: 7.9 Å (15% effective packing)
• Includes space for H-bond network (avg. bond length: 2.5 Å)
```

We can see that a length of 5.0 Å is not ideal for this system, therefore we can either change it directly in the input file or instruct the Ascec script to use a suggested length and rewrite the parameter itself.

```
ascec w6.asc r3 --box10
```

*The r3 command tells the script to run the annealing in triplicate, and the box10 flag uses a box length for an effective packing density of 10%.*

```
Using recommended box size: 9.1 Å (10.0% effective packing)
Creating 3 replicated runs in 'annealing/'...
  Created: w6_1/w6_1.asc
  Created: w6_2/w6_2.asc
  Created: w6_3/w6_3.asc
Created launcher script: launcher_ascec.sh

To run all simulations sequentially, use:
  ./launcher_ascec.sh
```

```
./launcher_ascec.sh
```

*A launcher is generated to run all replicas in series. Simply run it.*

```
(base) :~/.../examples/water_hexamer/orca6$ ./launcher_ascec.sh
ASCEC v04 environment is now active via direct script setup.
```

```
    * ASCEC-v04: Feb-2026 *

    Using random seed: 115439

    Attempting initial QM energy calculation...

    Attempt 1/100 for initial QM calculation.
    Calculation successful. Energy: -71.61760217 a.u.

    Using rigid-body moves only (translation + rotation)

    Starting annealing simulation...


    Annealing Step 1/100 at Temperature: 600.00 K

    Annealing Step 2/100 at Temperature: 570.00 K

    (...)

    Annealing Step 99/100 at Temperature: 3.94 K

    Annealing Step 100/100 at Temperature: 3.74 K

    Annealing simulation finished.

    Final lowest energy found: -71.65131472 a.u.
    Total QM calculations performed: 2087
    ================================================================
```

*The process starts from a "Big Bang" position, where the centers of mass of all molecules are placed overlapping at the center of the simulation box. This high-energy state represents the worst possible starting condition, which helps in avoiding biases in the trajectory. To resolve this situation, random movements are made until a stable configuration is found. A configuration is considered successful when its potential energy can be calculated without errors, such as those arising from atomic overlaps.*

*The system will make up to 100 attempts to find such a state. The first successful configuration becomes the starting point from which the annealing simulation begins. Should all 100 attempts fail, the simulation will terminate, providing data that can be used to diagnose the problem or refine the initial parameters.*

## 11.2 Optimization Strategy

### Preprocessing Annealing Candidates

The stochastic annealing protocol generates a vast ensemble of redundant configurations, making immediate high-level quantum mechanical refinement computationally prohibitive. To address this bottleneck, a pre-optimization stage is implemented using the efficient semi-empirical **GFN2-xTB** method [7]. Crucially, this intermediate level of theory must maintain the topological fidelity of the Potential Energy Surface (PES), ensuring that geometries relax into their nearest local minima without erroneously merging distinct basins or eliminating valid low-energy isomers. To validate the efficacy of this strategy and strictly balance computational efficiency with accuracy, this work evaluates two distinct protocols: a hierarchical two-step refinement, and a benchmark approach involving the direct processing of the raw annealing data.

```
/
├── annealing/
├── b97_orca.inp
├── gfn2_orca.inp
├── launcher_orca.sh
└── w6.asc
```

### 11.2.1 Input Templates

**I. ORCA Input Template (`gfn2_orca.inp`)**
This is a standard ORCA input file with specific placeholders that ASCEC utilizes to inject configuration data. In ORCA, lines starting with `#` are comments.

- `#name`: (Optional) ASCEC replaces this with the filename for traceability.
- `#`: A single hash mark tells the script where to insert the Cartesian coordinates block.

```
#name
! Opt GFN2-xTB TightSCF Numfreq PAL8
%geom
  maxiter 5000
end

* xyz 0 1
#
*
```

**II. Launcher Template (`launcher_orca.sh`)**
This shell script defines the environment variables and execution paths for the quantum chemistry software. It includes a specific placeholder (`###`) where ASCEC will append the sequence of execution commands for all generated input files.

```
#!/bin/bash

# Define paths to ORCA 6.1.1 installation
export ORCA_BASE=$HOME/software
export ORCA611_ROOT=$ORCA_BASE/orca_6_1_1
export OPENMPI418_ROOT=$ORCA_BASE/openmpi-4.1.8

# Save system paths to prevent nesting
_SYSTEM_PATH="$PATH"
_SYSTEM_LD_LIBRARY_PATH="$LD_LIBRARY_PATH"

# Export ORCA and OpenMPI paths
export PATH="$ORCA611_ROOT:$OPENMPI418_ROOT/bin:$_SYSTEM_PATH"
export LD_LIBRARY_PATH="$ORCA611_ROOT:$OPENMPI418_ROOT/lib:
$_SYSTEM_LD_LIBRARY_PATH"

echo "ORCA 6.1.1 environment is now active."
mpirun --version

###
```

### 11.2.2 Optimization Setup

**Generating Optimization Files**
The ASCEC `opt` command scans the `annealing` directory for result files (`.xyz`) and interactively guides the user to generate the necessary input files for pre-optimization.

```
ascec opt gfn2_orca.inp launcher_orca.sh
```

```
(base) :~/.../water_hexamer/orca6$ ascec opt gfn2_orca.inp launcher_orca.sh
Found 3 XYZ file(s) to process:
- ./annealing/w6_1/result_115439.xyz
- ./annealing/w6_2/result_609695.xyz
- ./annealing/w6_3/result_154701.xyz

============================================================
XYZ file selection
============================================================

Special options:
a. Process all result_*.xyz files independently
c. Combine all result_*.xyz, then process the combined file

Select files (numbers separated by spaces, 'a' for all, or 'c' to combine): c

Merged 3 files into optimization/combined_r3.xyz with 416 configurations
  Created: opt_conf_1.inp
  Created: opt_conf_2.inp
  (...)
  Created: opt_conf_416.inp

Completed processing. Created 416 input files total.
Created calculation system in 'optimization' directory.
```

**Resulting Input Files**
The script automatically injects the coordinate data and comments into the templates. Note how the placeholders have been replaced:

```
# Configuration: 1 | E = -71.61760217 a.u. | result_115439

! Opt GFN2-xTB TightSCF Numfreq PAL8

%geom
  maxiter 5000
end

* xyz 0 1
O       7.521687      8.123473      2.987348
H       7.729738      7.604909      3.779022
(...)
*
```

```
#!/bin/bash
# ... (Environment variables as defined in template) ...

echo "ORCA 6.1.1 environment is now active."

###

# Run QM using the full path
$ORCA611_ROOT/orca opt_conf_1.inp > opt_conf_1.out ; \
$ORCA611_ROOT/orca opt_conf_2.inp > opt_conf_2.out ; \
(...)
$ORCA611_ROOT/orca opt_conf_416.inp > opt_conf_416.out
```

**Executing the Optimizations**

Navigate to the newly created `optimization` directory and execute the launcher script.

```
cd optimization
./launcher_orca.sh
```

### 11.2.3 Result Organization

**Sorting and Summarizing**

Upon completion of the optimizations, the directory will be populated with thousands of output files. The `sort` command organizes these files into subfolders (`inputs`, `outputs`, `xyz`) and generates a summary of the computational wall time.

```
ascec sort
```

```
(base) :~/.../water_hexamer/orca6/optimization$ ascec sort
================================================
ASCEC Sort Process Started
================================================

1. Sorting files by base names...
```

```
(...)
Processing: ./opt_conf_415/opt_conf_415.xyz
Processing: ./opt_conf_416/opt_conf_416.xyz

4. Creating calculation summary...
Processing 416 ORCA files in parallel...
Completed processing 416 ORCA files successfully.
Summary written to orca_summary.txt

5. Collecting .out files...
Copied 416 .out files to cosmic_2/orca_out_416
```

*Note: If the summary is not required (e.g., to save time on I/O operations), use the* `--nosum` *flag to perform a quick sort only.*

```
head -n 20 summary_walltime.txt
```

```
Summary of all optimization:
Number of jobs: 416
Total execution time: 4:6:33.611
Mean execution time: 0:0:35.562
Shortest execution time: 0:0:19.172
Longest execution time: 0:1:39.882
Total wall time: 4 hours, 6 minutes

=> 1. opt_conf_400.out
energy = -30.49184195291
time = 0:0:19.172

=> 2. opt_conf_272.out
energy = -30.49391845293
time = 0:0:19.831
```

Listing 1: Example of Pre-optimization Summary

*The pre-optimization of 416 structures was completed in approximately 4 hours, which is a reasonable time with room for improvement.*

## 11.3 COSMIC Clustering

### 11.3.1 Configuration Screening

The `sort` command (executed previously) extracts all valid output files (e.g., `.out` or `.log`) into a dedicated subdirectory. This prepares the dataset for the topological analysis.

```
/
├── annealing/
├── calculation/
└── cosmic/
    └── orca_out_416/
        ├── opt_conf_1.out
        ├── ...
        └── opt_conf_416.out
```

```
└─ w6.asc
```

To initiate the topological analysis, navigate to the `cosmic` directory. The script is executed using the `cosmic` command, which accepts two primary control arguments:

- `--th` **(Threshold):** Defines the clustering granularity. This value represents the maximum Euclidean distance in the physicochemical feature space for two structures to be considered part of the same motif.
- `-j` **(Jobs):** Specifies the number of CPU cores to use for parallel parsing of the output files. If not specified, the script will attempt to use all available cores.

**Understanding the Threshold:**
A lower threshold creates tighter, more homogeneous clusters (distinguishing subtle geometric variations), while a higher threshold yields broader, more inclusive families. The choice of threshold depends on the system size and the desired resolution:

| Granularity | Value ($\tau$) | Typical Use Case |
|---|---|---|
| **Strict / Tight** | `0.5 - 1.5` | **Small/Rigid Systems.** Distinguishes between specific rotamers or iso-energetic conformers with slight dihedral changes. |
| **Standard** | `2.0 - 2.5` | **General Purpose.** Default for molecular clusters (e.g., water hexamers). Balances distinct topology against vibrational noise. |
| **Loose / Broad** | `> 3.0` | **Large/Flexible Systems.** Groups broad structural families (e.g., folding patterns) while ignoring local side-chain fluctuations. |

Table 7: Recommended clustering threshold ($\tau$) values.

### 11.3.2 Automatic Threshold Selection

The values in Table 7 are useful manual overrides, but by default COSMIC does **not** require the user to pick $\tau$. In `automatic` mode the threshold is detected *per case* from the distribution of merge heights in the UPGMA [11] linkage matrix. This matters because a single fixed value is not universally appropriate: for atomic clusters and weakly bound (dispersion- or van der Waals-dominated) systems the per-feature variance is intrinsically small, so almost every merge falls below $\tau = 2.0$ and physically distinct motifs collapse into a few mega-clusters. A data-driven cut fixes this.

**Knee detection (default).** Let $h_{(1)} \le h_{(2)} \le \cdots \le h_{(n-1)}$ be the sorted merge heights of the linkage matrix for $n$ input structures ($n-1$ merges). Plotted against merge index, this curve stays nearly flat while structures within the same motif fuse, then rises sharply once distinct motifs begin to merge; the elbow marks that transition and is the natural place to cut. Since the index and the height live on unrelated scales, both axes are first mapped to the unit interval:

$$\tilde{x}_i = \frac{i-1}{n-2}, \qquad \tilde{y}_i = \frac{h_{(i)} - h_{(1)}}{h_{(n-1)} - h_{(1)}}, \qquad i = 1, \ldots, n-1. \tag{15}$$

The perpendicular distance from the normalised point $(\tilde{x}_i, \tilde{y}_i)$ to the secant $\tilde{y} = \tilde{x}$ joining the first and last merges is

$$d_i = \frac{\left|\tilde{x}_i - \tilde{y}_i\right|}{\sqrt{2}}, \tag{16}$$

and the elbow is the merge of maximal deviation,

$$i^* = \arg\max_i d_i. \tag{17}$$

The cut is then placed *midway* between the elbow merge and the next one,

$$\tau_{\text{auto}} = \tfrac{1}{2}\big(h_{(i^*)} + h_{(i^*+1)}\big), \tag{18}$$

so the elbow merge itself is still treated as within-cluster and only merges in the steep regime act as separators. This is the standard maximum-distance-to-secant ("kneedle") heuristic [12]; COSMIC uses a pure NumPy implementation to avoid an extra dependency.

**Sanity gates and the $\tau = 2.0$ cap.** The knee is trusted only when three conditions hold; otherwise COSMIC falls back to the empirical fixed threshold $\tau = 2.0$:

1. at least 8 merges ($n - 1 \geq 8$) — smaller datasets lack the curve structure for a reliable elbow;
2. a non-trivial dynamic range, $h_{(n-1)}/h_{(1)} \geq 5$ — otherwise the curve is essentially a straight line and the elbow is meaningless;
3. a sensible cluster count, $2 \leq k_{\text{auto}} \leq \lfloor n/2 \rfloor$ — guarding against one giant cluster or pathologically many.

In addition, the empirical value acts as an **upper limit**: if the detected knee exceeds 2.0, the threshold is capped at $\tau = 2.0$. The source of the finally applied threshold (`knee`, `legacy`, or `user`) is reported in the verbose log and on the diagnostic plot.

**The empirical $\tau = 2.0$ value (the $2\sigma$ rule).** This was the historical default and remains available as an explicit override (`--threshold 2.0`) and as the fallback above. Its justification is statistical. For two $z$-standardised feature vectors the squared Euclidean distance is $d^2 = n_c\,\overline{\Delta z^2}$, so the root-mean-square per-feature difference at the cut is $\Delta z_{\text{rms}} = \tau/\sqrt{n_c}$, measured directly in standard-deviation units:

| Features ($n_c$) | $\Delta z_{\text{rms}}$ at $\tau = 2.0$ | Interpretation |
|---|---|---|
| 5 | $0.89\sigma$ | per feature $\lesssim 1\sigma$ |
| 7 | $0.76\sigma$ | per feature $\lesssim 1\sigma$ |
| 11 | $0.60\sigma$ | per feature $< 1\sigma$ |
| 14 | $0.53\sigma$ | per feature $\approx 0.5\sigma$ |

Table 8: Per-feature RMS difference corresponding to $\tau = 2.0$ for typical feature-vector sizes $n_c$.

In every case $\tau = 2.0$ marks the boundary where the average per-feature difference crosses from *statistically indistinguishable* ($< 1\sigma$) to *statistically distinguishable* ($\geq 1\sigma$): structures merged below it share a common physicochemical profile within noise, while those separated above it differ meaningfully. The same boundary can be read as a Pearson similarity [13] between the two profiles, the basis of the per-cluster similarity floor reported by COSMIC.

**Why Mojena is not used.** The Mojena stopping rule [14] is a well-known clustering criterion and is still computed by COSMIC, but only as a diagnostic line on the validation plots — it is *not* part of the fallback chain. Across many independent test cases it proved too inconsistent to drive cluster selection reliably, which is why the knee method (with the $\tau = 2.0$ fallback) is preferred.

### 11.3.3 Execution and Directory Selection

To initiate the analysis, navigate to the `cosmic` directory. The script offers two primary modes of execution:

**A. Standard Physicochemical Clustering**
Uses the feature vector approach to group structures based on electronic and energetic properties.

```
cd cosmic/
cosmic --th=2 -j4
```

*In this example, a standard threshold of $\tau = 2.0$ is applied using 4 CPU cores.*

**B. Hierarchical Two-Step Refinement**
Activates the secondary geometric check.

```
cosmic --th=2 --rmsd=1 -j4
```

*This command executes the hierarchical protocol. First, the algorithm performs the standard physicochemical screening ($\tau = 2.0$). Subsequently, it applies a geometric refinement within each identified family, subdividing structures that exceed a Root Mean Square Deviation (RMSD) of 1.0 Å. Used to distinguish stereoisomers or iso-energetic conformers*

**Interactive Selection**
Upon execution, the script detects available directories containing output files. Select the target folder by entering its corresponding index number.

```
Found the following folder(s) containing quantum chemistry log/out files:

[1] orca_out_416 (Contains: .out)

Enter the number of the folder to process, or type 'a' to process all: 1

Extracting data from 416 files...
Using 4 CPU cores for parallel processing
Data extraction complete. Proceeding to clustering.
```

```
Clustering 416 configurations...

Cluster 1 (31 configurations)
Cluster 2 (11 configurations)
Cluster 3 (22 configurations)
...
```

### 11.3.4 Performance Optimization: Data Caching

The parsing of hundreds of quantum chemical log files is an I/O-intensive process that constitutes the primary wall-time bottleneck. To mitigate this, the script implements a **serialization mechanism**. Upon the first successful run, the parsed physicochemical descriptors are saved into a binary cache file (e.g., `data_cache_xxxxxx.pkl`).

```
/
├── orca_out_416/
└── data_cache_203971.pkl
```

**Using the Cache**
For subsequent analyses (e.g., testing different clustering thresholds), the script automatically detects and loads the binary cache, bypassing the heavy parsing stage entirely.

```
Attempting to load data from cache: 'data_cache_203971.pkl'
Using cached data
Data extraction complete. Proceeding to clustering.
(...)
```

**Forcing Reprocessing**
If the underlying output files have been modified or added to, the cache must be updated. Use the `--reprocess-files` flag to do this, or simply delete the cache.

```
cosmic --th=2 -j4 --reprocess-files
```

### 11.3.5 Cluster Representatives and Motifs

The algorithm groups structurally similar configurations based on physicochemical feature vectors. Within each identified cluster, the structure with the lowest potential energy is selected as the representative configuration for that cluster and marked as a unique motif.

```
Creating 50 motifs from cluster representatives...
Created motifs dendrogram: motifs_dendrogram.png
Motifs created: 50 representatives saved to motifs_50

Boltzmann distribution saved to 'boltzmann_distribution.txt'
Clustering summary saved to 'clustering_summary.txt'

Finished processing folder: orca_out_416

All selected molecular analyses complete!
```

### 11.4 Final Optimization Setup

**Generating High-Level Input Files**
Once the unique motifs have been identified, they typically require refinement at a higher level of theory (e.g., DFT with larger basis sets). The ASCEC `opt` command streamlines this process.

Similar to the `calc` step, this command scans the directory for coordinate files. However, instead of processing raw annealing results, it targets the **pre-processed configurations**.

```
ascec opt b97_orca.inp launcher_orca.sh
```

```
(base) :~/.../water_hexamer/orca6$ ascec opt b97_orca.inp launcher_orca.sh

============================================================
Optimization XYZ file selection
============================================================

Combined files:
0) ./calculation/combined_results.xyz
1) ./calculation/combined_r3/combined_r3.xyz
2) ./cosmic/motifs_53/all_motifs_combined.xyz

Motif files:
3) ./cosmic/motifs_53/motif_01_opt_conf_228.xyz
4) ./cosmic/motifs_53/motif_02_opt_conf_119.xyz
5) ./cosmic/motifs_50/motif_03_opt_conf_91.xyz
```

*Select **Option 2** (`all_motifs_combined.xyz`) to create input files for the 53 detected motifs. Submit these files for higher-level optimization (e.g., B97-3c), then perform a second `sort` (creating the `cosmic_2` directory) and COSMIC analysis. This workflow—advancing from raw configurations to preliminary motifs, and finally to optimized unique motifs—is required to obtain an accurate Boltzmann distribution.*

## 12 Output Files

A successful run of ASCEC with the annealing option ("1" in Line 1) with candidate structure clustering using the COSMIC script for this example protocol will generate several files.

### 12.1 Annealing output

#### 12.1.1 Main Log File (w6.out)

This file serves as the primary log, recording the complete history of the annealing protocol. To verify that the simulation completed successfully without runtime errors, users should inspect the final lines of the file (e.g., via the command `tail w6.out`). A successful run is confirmed by the presence of the termination flag:

```
** Normal annealing termination **
```

```
tail -n 15 w6_1.out
```

```
============================================================

** Normal annealing termination **
Total Wall time: 0 days 0 h 30 min 52 s 880 ms
Energy was evaluated 2919 times

Energy evolution in tvse_115889.dat
Configurations accepted by Max.-Boltz. statistics = 72
Accepted lower energy configurations = 55
Accepted configurations in result_115889.xyz = 127
Lowest energy configuration in rless_115889.out
Lowest energy = -0.51829172 u.a. (Config. 127)

============================================================
```

### 12.1.2 Reproducibility and Metadata

To ensure scientific rigor, ASCEC generates essential metadata for data validation and traceability. Each annealing run is initialized with a unique 6-digit random seed, which is used to seed all internal pseudorandom number generators (PRNGs). This mechanism ensures the stochastic independence of parallel runs, preventing statistical bias in the exploration of the potential energy surface. The seed is logged in the output file and appended to the names of key output files (e.g., result_957775.xyz, tvse_957775.dat), enabling clear identification and organization of results. Rerunning ASCEC with the same seed and input parameters will reproduce the same sequence of proposed molecular moves. However, exact reproduction of the full annealing trajectory additionally requires an identical software environment — the same Python and NumPy versions, as well as the same version, compilation, and parallelization settings of the external quantum-chemistry program (e.g., ORCA or xTB) — since numerical differences in energy evaluations, however small, can alter accept/reject decisions and cause trajectories to diverge.

### 12.1.3 Simulation Generated Data

**I. Global Minimum Candidate (`rless_xxxxxx.out`)**
This file contains the Cartesian coordinates corresponding to the lowest energy configuration visited during the entire annealing run. The suffix `xxxxxx` corresponds to the unique random seed of the run.

```
# Configuration: 141 | Energy = -71.65131472 u.a.
3
water1
O       3.114949    -2.572173     0.846802
H       3.740289    -3.084485     1.381054
H       2.849619    -3.185249     0.144893
3
water2
O      -1.728986     1.116733    -0.116378
H      -2.554571     0.817409    -0.525959
(...)
```

**II. Energy Evolution Log (`tvse_xxxxxx.dat`)**
This dataset records the thermodynamic history of the simulation. It is formatted in three columns: **Step Number**, **Temperature (K)**, and **Potential Energy (Hartree)**. Data is recorded exclusively for accepted Monte Carlo steps. This file is the source for generating the energy profile plots (see Figure 19) and the aggregate replica comparison (see Figure 20). These diagrams are automatically produced by the ASCEC script. To regenerate them manually, use the `diagram` command with the `--scaled` flag to auto-scale the Y-axis.

```
ascec diagram --scaled
```

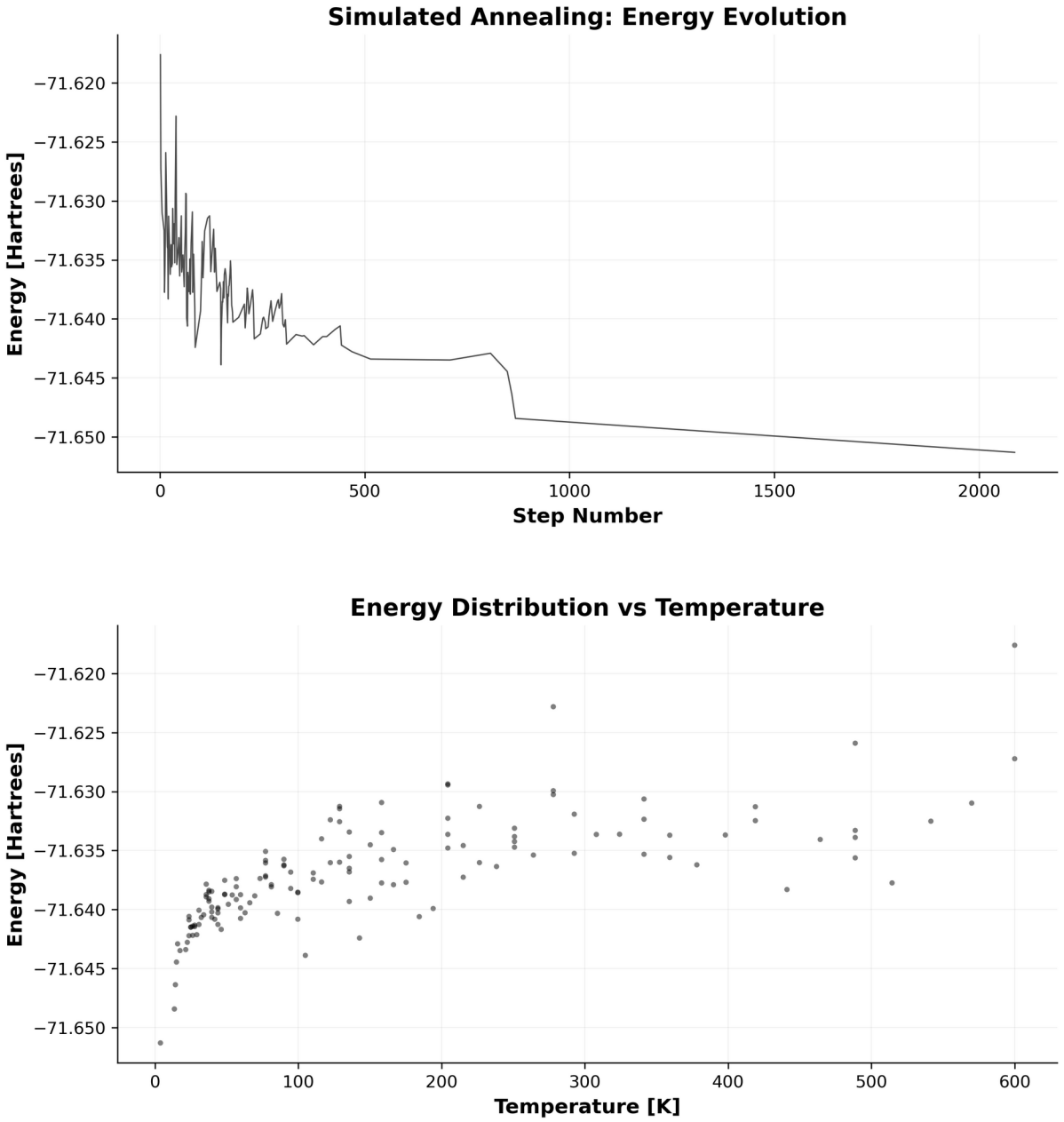



Figure 19: Single-run energy evolution profile (Temperature vs. Energy) generated by ASCEC.

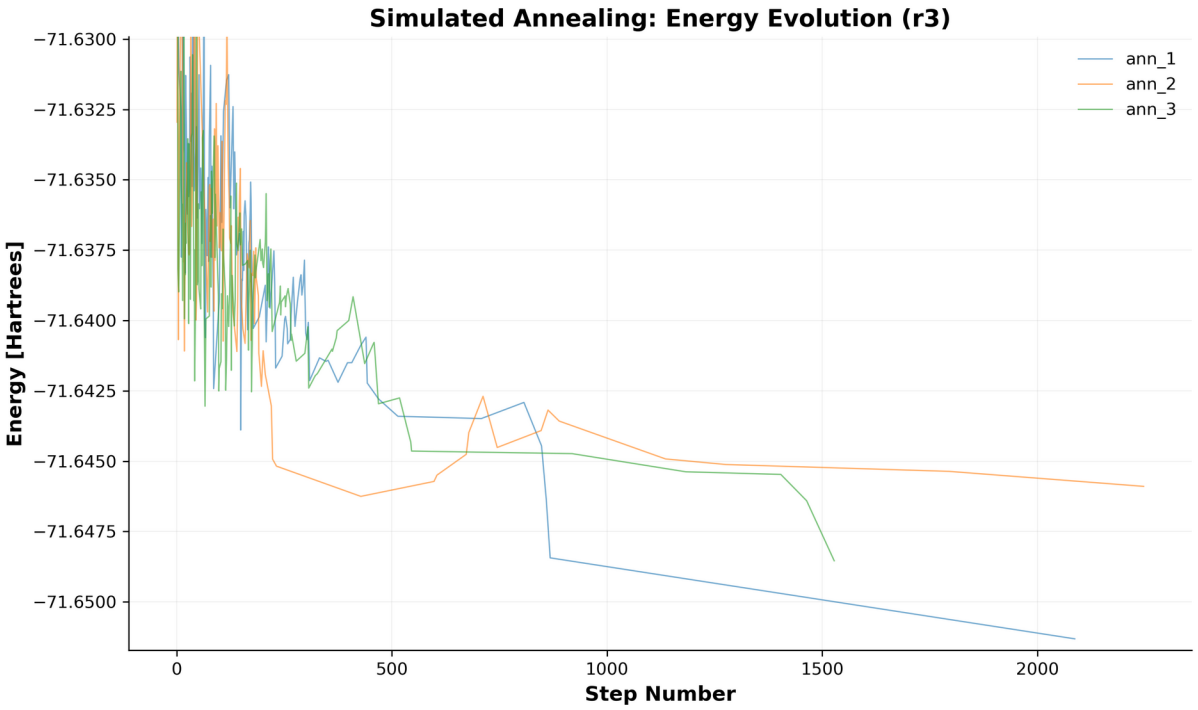



Figure 20: Combined energy profiles comparing multiple replica executions.

**III. Structural Trajectory** (`result_xxxxxx.xyz` | `.mol`)
These files contain the Cartesian coordinates of the entire ensemble of accepted structures. They serve as the full trajectory of the simulation, formatted for immediate visualization and animation using standard molecular editors (e.g., Avogadro2, Molekel, IQmol, or GaussView).

```
18
Configuration: 1 | E = -71.61760217 a.u. | T = 600.0 K
O       7.521687     8.123473     2.987348
H       7.729738     7.604909     3.779022
(...)
18
Configuration: 2 | E = -71.62722332 a.u. | T = 600.0 K
O       7.649736     8.853069     2.485145
H       8.312217     8.157098     2.610376
18
Configuration: 3 | E = -71.63098312 a.u. | T = 570.0 K
O       8.358924     7.910310     1.934682
H       9.138323     7.334893     1.954008
```

**IV. Simulation Box Visualization** (`resultbox_xxxxxx.xyz` | `.mol`)
Unless the `-nobox` flag is used, the program generates auxiliary files containing the simulation box vertices as dummy atoms. This allows for the visual verification of the confinement boundaries relative to the molecular system.
*Compatibility Note:*

- **Avogadro2:** Supports both `.xyz` and `.mol` formats for box visualization.
- **IQmol & GaussView:** Require the `.mol` file to correctly render the dummy atoms.

## 12.2 Calculation & Optimization Output

Once the process of calculating the generated input configurations and their sorting has been completed, the working directory (e.g., `calculation/` or `optimization/`) will typically contain the following key components:

- **Master Trajectories (`combined_results.xyz | .mol`):** Single master files containing the aggregated Cartesian coordinates for all processed configurations. These are generated to allow for the immediate visualization of the entire ensemble.

```
18
Coordinates from ORCA-job opt_conf_1 E -30.494666183120
O           4.89755049315488      6.57402652599668      3.11667076076455
H           4.49984221242737      6.65106539506547      4.01165830268415
(...)
18
Coordinates from ORCA-job opt_conf_2 E -30.489323286040
O           5.03567346983380      6.71360166724240      4.08623792011938
H           4.16925086054156      7.14879619512198      4.15686651181911
H           4.91370051289682      6.02838117620930      3.40385389606381
```

- **Job Subdirectories:** Dedicated folders containing the specific input/output files for each.
  - *In Calculation:* Naming depends on the input generation strategy:
    - **Option 'c' (Combine):** `opt_conf_#` (e.g., `opt_conf_1`). Numbering is sequential across the merged dataset.
    - **Option 'a' (Separate):** `opt#_conf_#` (e.g., `opt1_conf_1`). Explicitly identifies the source replica (e.g., Configuration 1 from Replica 1).
  - *In Optimization:* Named `motif_#` (e.g., `motif_01`).
- **Execution Summary (`*_summary.txt`):** A text file detailing the computational duration (wall time) and final energies for each calculation.
- **Launcher Script:** The shell script used to execute the batch jobs (e.g., `launcher_orca.sh`).

Exclusively in the **Calculation** directory, if the **Combine** strategy (Option `c`) was selected during setup, a file named `combined_r#` (e.g., `combined_r3`) is generated. This holds the raw merged data from the annealing replicas that were used to generate the initial input files.

## 12.3 COSMIC Output

Once the clustering process concludes, the results are stored in a dedicated directory. Note that the directory creation is driven by the `sort` command, which automatically increments the suffix (e.g., `cosmic_2`) to prevent overwriting. A typical output structure contains:

```
/
├── dendrogram_images/
├── extracted_clusters/
├── extracted_data/
├── motifs_53/
└── orca_out_416/
```

```
├── skipped_structures/
├── boltzmann_distribution.txt
├── clustering_summary.txt
└── data_cache_105638.pkl
```

### 12.3.1 Dendrograms

The dendrogram is the main visualization tool for understanding the topology of the sampled set. It represents a hierarchical tree in which the vertical axis (Y) indicates the **Euclidean distance** (dissimilarity) between configurations.

By default, a single dendrogram (`dendrogram.png`) is generated for the entire dataset. The optional `--group-hb` flag produces per-H-bond-family dendrograms (`dendrogram_H{N}.png`), which can be useful for visual inspection of specific interaction networks. The examples below use `--group-hb` to illustrate family-specific trees.

**Understanding the Clustering Logic (Figure 21)** The structure of the dendrogram directly informs the choice of the clustering threshold ($\tau$). As illustrated in the specific case of an H-Bond=10 family (using `--group-hb`), where three clusters were identified:

1. **Redundancy Identification:** Configurations **30** and **31** merge at a low height ($d < 1.0$). This implies they are geometrically and electronically equivalent.
2. **Cluster Differentiation (Intermediate):** Configuration **51** joins the (30+31) pair at a distance of $d \approx 3.5$. With a standard threshold of $\tau = 2.0$, this structure falls **above** the cutoff. Consequently, it is **not** grouped with the pair and is identified as a distinct motif.
3. **Global Separation:** Configuration **48** joins the ensemble at a much higher distance ($d \approx 5.8$). Since $5.8 \gg \tau$, it is recognized as a completely distinct basin of attraction, indicating significant structural and electronic differences.

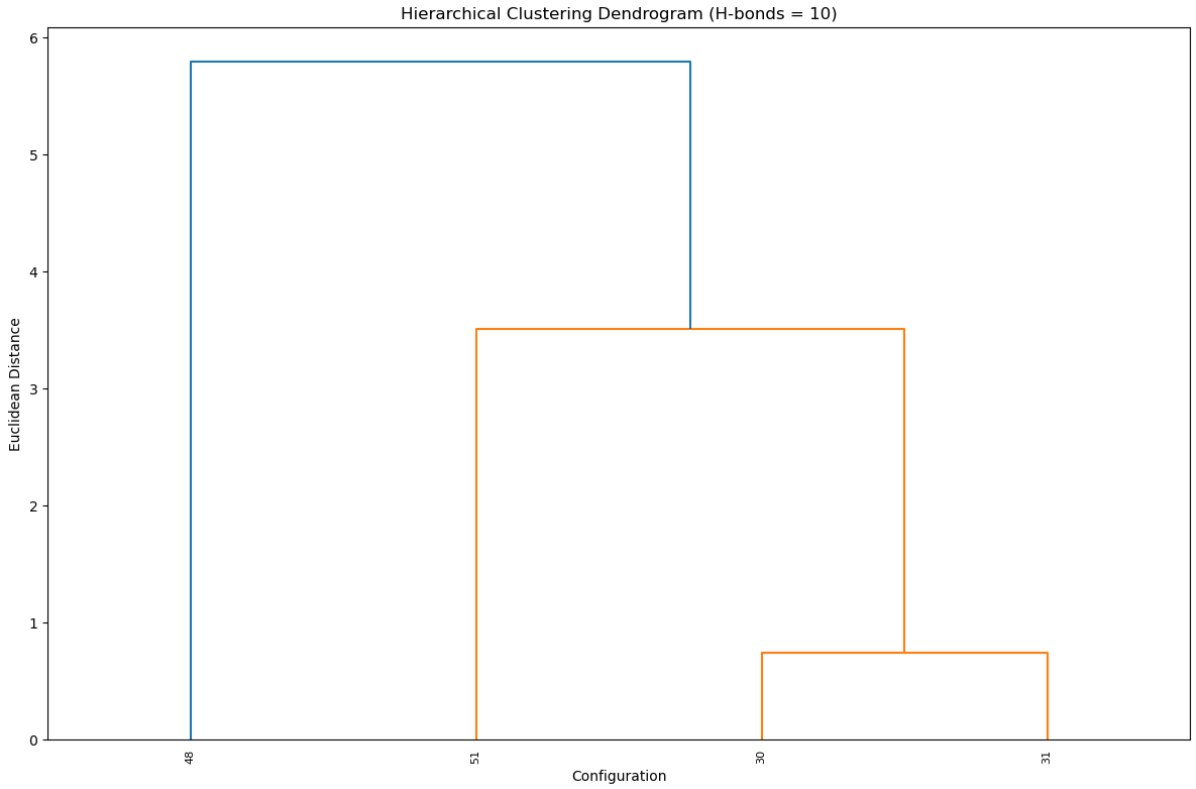



Figure 21: Detailed view of a dendrogram branch for the H-bonds=10 family (generated with `--group-hb`). The vertical height of horizontal nodes indicates the dissimilarity between branches.

**Family-Specific Trees (`--group-hb`):** When the `--group-hb` flag is active, the COSMIC module splits the dataset based on the number of detected Hydrogen Bonds (HB), generating a separate dendrogram for each subset. This allows granular inspection of specific interaction networks. Without this flag, all structures appear in a single unified dendrogram.

**Pre-Optimization Analysis (GFN2-xTB)** Figure 22 displays the clustering results for the H-Bond=9 family following the initial semi-empirical pre-optimization (using `--group-hb`).

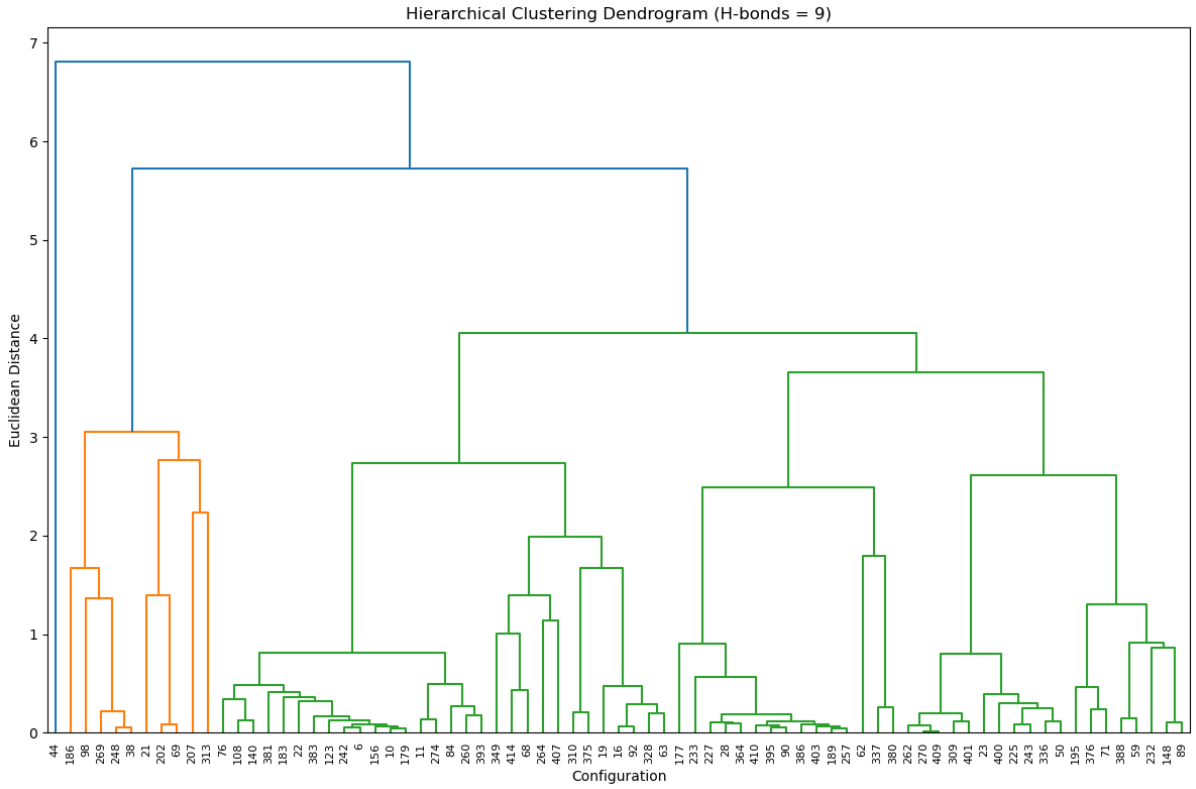



Figure 22: Dendrogram for the H-Bond=9 family following GFN2-xTB pre-optimization (generated with `--group-hb`).

As shown in Figure 22, the tree exhibits a high density of leaves with deep, complex branching. The presence of distinct major families at large Euclidean distances indicates that the annealing process successfully located fundamentally different structural basins. However, the dense grouping at the bottom of the clusters suggests a high degree of redundancy.

**Hierarchical Reduction Strategy:**
In this specific example, the preprocessing step successfully reduced the ensemble from **416 raw candidates** to **53 motifs**. These representatives were subsequently re-optimized using a higher level of theory (B97-3c), followed by a secondary clustering analysis which yielded **20 unique motifs**. It is crucial to note that the final number of motifs diverges from the intermediate count, as the topology of the Potential Energy Surface (PES) is sensitive to the electronic structure method employed; distinct semi-empirical minima may merge into a single basin upon DFT refinement. Furthermore, for this example, we did not correct the structures that the script flagged as critical and requiring manual inspection, so this final figure could be slightly higher.

- Impact of critical structures on total dataset: 8/416 configurations (1.9%)
- Impact of critical structures on total dataset: 4/53 configurations (7.5%)

**Final Optimization Analysis (DFT/B97-3c)** Figure 23 presents the topological classification of the H-Bond=7 family after high-level refinement.

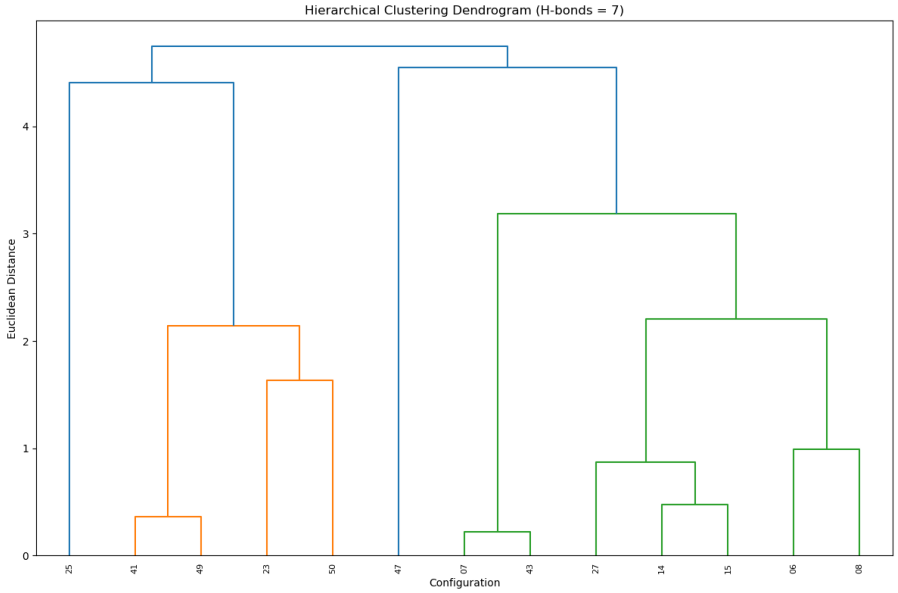



Figure 23: Dendrogram for the H-Bond=7 family following B97-3c final optimization (generated with `--group-hb`).

As shown in Figure 23, the topology is significantly more structured than the pre-optimization stage. The clear vertical separation between nodes confirms that the high-level optimization has relaxed the structures into well-defined, distinct minima, facilitating the selection of a robust threshold for the final Boltzmann population analysis.

**Motifs Dendrogram** Figure 24 presents the topological classification of the motifs found.

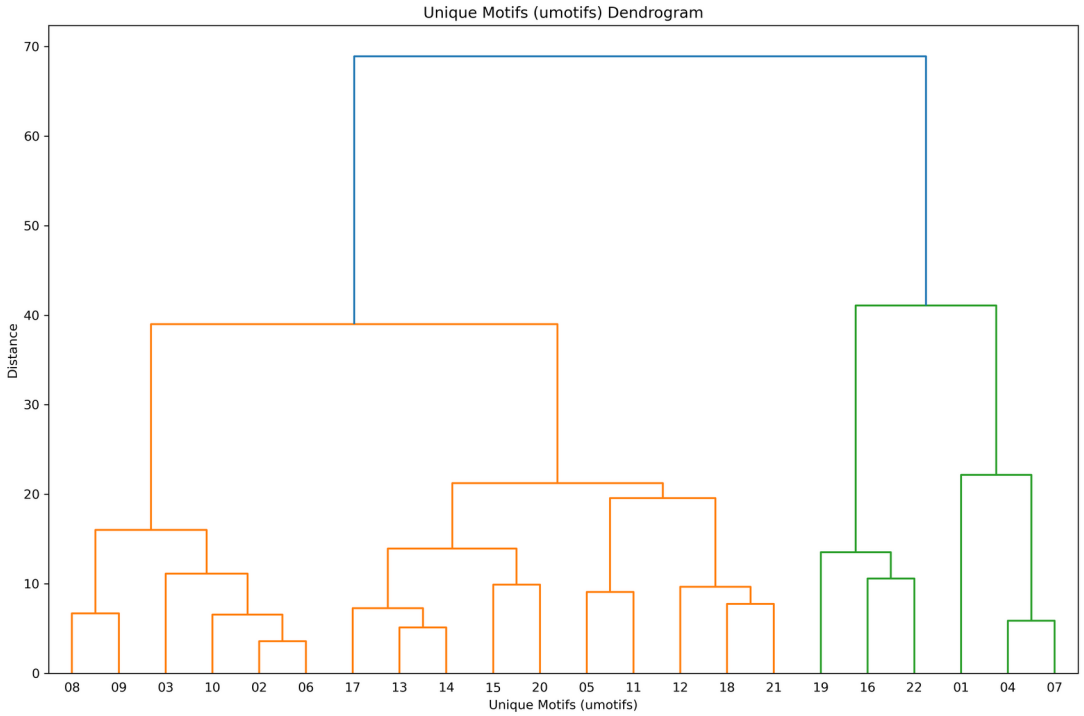



Figure 24: Dendrogram for the 22/23 unique motifs found after final optimization using B97-3c.

### 12.3.2 Clustering Summary Log

The script generates a comprehensive summary file (`clustering_summary.txt`) containing global statistics and detailed cluster breakdowns.

```
Clustering Results for: orca_out_416
COSMIC threshold (distance): 2.0
Total configurations processed: 416
Total files skipped: 23 (5.5%)
Critical skipped files: 8 (1.9%)
Total number of final clusters: 53
```

**Example of Non-Converged Output** When certain structures did not converge, the summary report indicates "N/A" for Gibbs energy. This indicates that the associated ORCA jobs reached the maximum number of optimization cycles without converging, so the thermochemical calculation step was not performed. Even so, the script was able to analyze the last Cartesian coordinates and attempted to cluster them.

```
All configurations

Configurations: 53
Number of clusters: 23

Cluster 22 (3 configurations):
Files:
- motif_25_opt.out (Gibbs Energy: N/A)
- motif_26_opt.out (Gibbs Energy: N/A)
- motif_31_opt.out (Gibbs Energy: N/A)

Cluster 23 (1 configurations):
Files:
- motif_42_opt.out (Gibbs Energy: N/A)
```

**Valid Cluster Analysis** For successfully converged systems, the summary provides precise thermodynamic data. Below is an example showing a clear separation between a singleton cluster (Cluster 6) and a grouping of two configurations (Cluster 5).

```
Cluster 5 (2 configurations):
Files:
- motif_23_opt.out (Gibbs Energy: -458.357020 Hartree (-287623.37 kcal/mol))
- motif_50_opt.out (Gibbs Energy: -458.356211 Hartree (-287622.87 kcal/mol))

Cluster 6 (1 configurations):
Files:
- motif_25_opt.out (Gibbs Energy: -458.356863 Hartree (-287623.27 kcal/mol))
```

*Note:* When using `--group-hb`, the summary groups results under "Hydrogen bonds: N" headers instead of "All configurations".

### 12.3.3 Error Handling and Diagnostics

The COSMIC module implements a comprehensive diagnostic protocol to ensure ensemble integrity. It classifies files based on termination status, convergence, and vibrational topology:

**1. Anomaly Classification** The algorithm detects three specific types of problematic structures, which can be found in the output files:

- **Execution Failures:** Files lacking the "Normal Termination" flag (e.g., due to crash).
- **Non-Converged Geometries:** Files that did not reach a stationary point within the maximum optimization cycles despite normal termination.
- **Topological Instabilities:** Structures containing **one or more imaginary frequencies**. This includes first-order saddle points (Transition States), higher-order instabilities (Hilltops), or soft modes (e.g., unrelaxed methyl rotations).

**2. Filtering Logic** To balance data retention with redundancy reduction, anomalies are processed via clustering:

- **Skipped Files:** Files that cannot be parsed or failed execution are automatically skipped. Additionally, structures with imaginary frequencies are skipped if they cluster tightly ($d <$ threshold) with a known true minimum, as these are considered "noisy" versions of a successfully identified basin.
  - **Flagged as Critical:** Any structure with imaginary frequencies that forms an **isolated cluster** (distinct from valid minima) is preserved and flagged as **Critical**. These are stored separately as they potentially represent:
    - Unique Transition States (TS).
    - Distinct local minimum that failed to converge purely due to numerical issues.
    - Valid basin with a flat potential surface (soft modes).

    *Action Required:* These files are exported for review and generally require manual inspection or re-optimization.

```
/
└── skipped_structures/
    ├── clustered_with_minima/
    │   ├── motif_26_opt.out
    │   └── motif_26_opt.xyz
    ├── need_recalculation/
    │   ├── combined_need_recalc.xyz
    │   ├── motif_11_opt.out
    │   └── motif_11_opt.xyz
    └── skipped_summary.txt
```

### 12.3.4 Generated Cluster Data

Beyond the summary, the script populates two key directories for detailed inspection:

1. `extracted_clusters/`: Contains subfolders for every identified cluster (e.g., `cluster_3_11` denotes Cluster #3 containing 11 configurations).

   Inside, the individual or combined (`.xyz | .mol`) files allow for visual superposition and verification of the cluster.

2. `extracted_data/`: Contains statistical data files (e.g., `cluster_15_4.dat`). These text files list the raw physicochemical descriptors (dipole moments, rotational constants, energies) for every member of the group, enabling quantitative analysis of the cluster's homogeneity, along with weighting and tolerances.

```
Cluster (represented by: cluster_15_4) (4 configurations)

- opt_conf_239.out
- opt_conf_287.out
- opt_conf_322.out
- opt_conf_53.out

Deviation Analysis (Max-Min / |Mean|):
Gibbs Free Energy (Hartree) %Dev: 0.00%
HOMO-LUMO Gap (eV) %Dev: 0.04%
Dipole Moment (Debye) %Dev: 2.80%
Radius of Gyration (Å) %Dev: 0.21%
Rotational Constant A (GHz) %Dev: 0.49%
Rotational Constant B (GHz) %Dev: 0.49%
Rotational Constant C (GHz) %Dev: 0.62%
Last Vibrational Frequency (cm^-1) %Dev: 0.02%
Number of Hydrogen Bonds %Dev: 0.00%

Clustering Weights Applied:
Gibbs Free Energy: 1.00

Clustering Absolute Tolerances Applied:
Gibbs Free Energy: 0.000005
```

### 12.3.5 Boltzmann Distribution

After optimization and clustering, the `boltzmann_distribution.txt` file is generated, ranking unique motifs by their Gibbs Free Energy at 298.15 K. While this reference temperature can be modified via the `-T=#` flag, the thermochemical properties must be recomputed accordingly to maintain accuracy. As illustrated below, the global minimum (UMotif 01) and the first local minimum (UMotif 02) are nearly iso-energetic, comprising more than 50% of the population.

```
Population by Energy Minimum
(assuming non-degeneracy, gi = 1)

Unique Motif (umotif) Assignment Summary
(sorted by Boltzmann population)

cluster_15 (umotif_01)
From structure: motif_03_opt
Gibbs Energy: -458.360815 Hartree (-287625.75 kcal/mol, -12472.63 eV)
Population: 26.77 %

cluster_1 (umotif_02)
From structure: motif_06_opt
Gibbs Energy: -458.360808 Hartree (-287625.75 kcal/mol, -12472.63 eV)
Population: 26.56 %
```

# 13 Workflow Mode

The automated protocol enables the execution of complex, multi-stage pipelines through a single directive. Specifically configured for the **ORCA** package. Unlike standalone execution—which requires user intervention to filter and resubmit files—the Workflow Mode autonomously manages the transition between annealing, optimization, and clustering. It includes built-in logic to correct problematic structures (e.g., imaginary frequencies), significantly reducing the manual overhead required to locate global minima.

## 13.1 Pipeline Logic

Figure 25 delineates the execution logic of the automated pipeline for a standard two-step refinement protocol. It illustrates the iterative progression from initial annealing to final refinement, including the automated feedback loops for correcting critical failures.

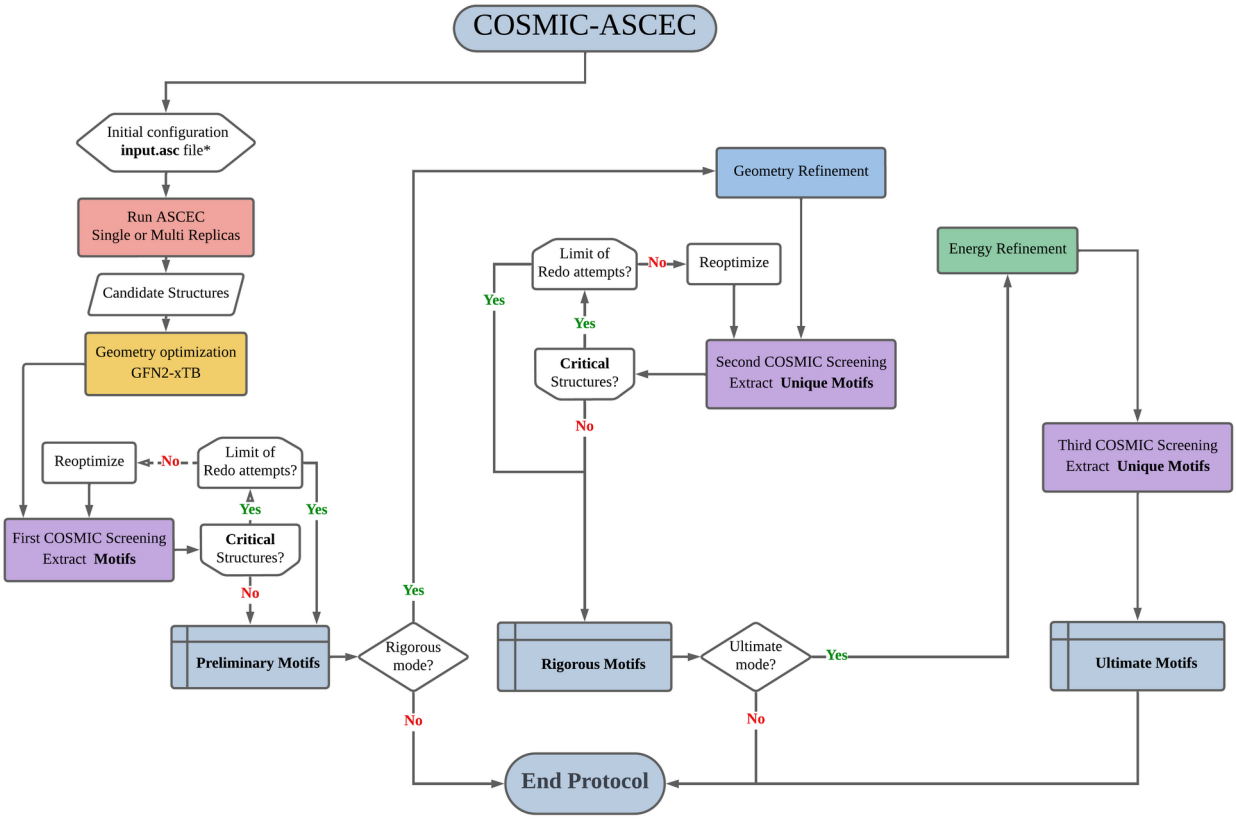



Figure 25: Schematic representation of the Two-Step Refinement Protocol.

## 13.2 Workflow Input (formic.asc)

To illustrate this functionality, we consider a conformational search for the formic acid dimer using **ORCA 6.1.1**. Starting from the high-energy *cis* dimer configuration, the workflow is tasked with automatically locating the experimentally observed *trans* global minimum. The following file represents the complete configuration, defining molecular parameters, simulation settings, and the multi-stage protocol.

```
# Input for formic acid dimer Simulation
```

```
1 10            # Line 1: Simulation Mode & Number of Config
8               # Line 2: Simulation Cube Length (Angstroms)
2               # Line 3: Annealing Quenching route
100.0 10.0 50   # Line 4: Linear Quenching Parameters
600.0 5.0 100   # Line 5: Geometric Quenching Parameters
3000  50        # Line 6: Maximum Monte Carlo Cycles per T and floor value
1.0 1.0         # Line 7: Maximum Displacement (A) & Rotation (radians)

30 60           # Line 8: Conformational sampling (%)
                # & Maximum dihedral rotation (degrees)
2 orca          # Line 9: QM Program Index & alias
pm3             # Line 10: Hamiltonian & Basis Set
1 8             # Line 11: nprocs (QM calculations and ASCEC evaluation)
0 1             # Line 12: Charge & Spin Multiplicity

2               # Line 13: Number of Molecules (nmo)

# Lines below: Molecule Definition

*
5
Formic_acid1
C    0.14624609581422     -0.35536580959883     -0.00000002457506
O    1.17429417535772      0.24095314517570      0.00000001235630
H    0.07493003024184     -1.46799267964298      0.00000000612430
O   -1.05240386842550      0.23715532959413     -0.00000000061204
H   -1.79206643298827     -0.38334998552802      0.00000000670651
*
5
Formic_acid2
C    0.14624609581422     -0.35536580959883     -0.00000002457506
O    1.17429417535772      0.24095314517570      0.00000001235630
H    0.07493003024184     -1.46799267964298      0.00000000612430
O   -1.05240386842550      0.23715532959413     -0.00000000061204
H   -1.79206643298827     -0.38334998552802      0.00000000670651
*

# Protocol
# (annealing, (pre)optimization, clustering, separated by "," or ".")

.asc,
r3 --box10,
calc --critical=0 --retry=20 --redo=10 ./preopt_gfn2.inp ./launcher_orca.sh,
cosmic --th=2 -j8,
opt --skipped=0 --retry=20 --redo=10 ./opt_wb97x.inp ./launcher_orca.sh,
cosmic --th=2 -j4
```

## 13.3 Protocol Command Syntax

The specific commands used in the `Protocol` section of the input file are detailed below.

| Command / Flag | Description |
|---|---|
| `.asc` | **Workflow Initiator.** Designates the file as a multi-stage workflow script. |
| `r3 -box10` | **Annealing.** Executes a triplicate ($r = 3$) annealing process. The `-box10` flag sets the effective packing box length to 10%. |
| `calc ... ./preopt_gfn2.inp` | **Pre-optimization.** Runs the preprocessing stage. |
| *–critical=0* | **Critical Tolerance.** Sets the allowance for "Critical" structures (isolated imaginary frequencies) to 0%, forcing the workflow to resolve all such cases. |
| *–retry=20* | **Execution Retries.** Automatically retries failed jobs (e.g., crashes) up to 20 times. |
| *–redo=10* | **Convergence Fixes.** Attempts to correct imaginary frequencies (e.g., saddle points) up to 10 times. |
| `cosmic -th=2 -j8` | **Clustering.** Groups structures using a COSMIC threshold of 2.0, utilizing 8 CPU cores (`-j8`) for parallel processing. |
| `opt ... ./opt_wb97x.inp` | **Optimization.** Performs high-level DFT optimization on the representative motifs identified in the previous stage. |
| *–skipped=0* | **Strict Convergence.** Sets the tolerance for "Skipped" (noisy/unconverged) files to 0%. Requires that all inputs successfully converge to valid minima. |
| `cosmic -th=2 -j4` | **Final Analysis.** Clusters the optimized structures to identify unique motifs and generate the final Boltzmann-weighted ensemble. |

Table 9: Description of commands and flags used in the automated workflow.

## 13.4 Execution and Outputs

The automated protocol is initiated via the following command:

```
ascec formic.asc protocol
```

### 13.4.1 Prerequisites and File Structure

Prior to execution, specific configuration and template files must be present in the working directory. These templates must adhere to the syntax defined in Section 11 (see 11.2.1), utilizing specific placeholders for dynamic data injection:

- `#name`: (Optional) Replaced by the specific filename to ensure traceability in output logs.
- `#`: Indicates the insertion point for the Cartesian coordinate block.
- `###`: Designates the location within the launcher script where serial execution commands will be injected.

Upon initialization, the system parses these files and generates a structured directory tree for each stage of the workflow.

`Annealing → Calculation → COSMIC → Optimization → COSMIC(2)`

| Required Input Files | Generated Output Structure |
|---|---|
| /<br>├── formic.asc<br>├── launcher_orca.sh<br>├── opt_wb97x.inp<br>└── preopt_gfn2.inp | /<br>├── annealing/<br>├── calculation/<br>├── optimization/<br>├── cosmic/<br>├── cosmic_2/<br>├── protocol_425319.pkl<br>└── protocol_summary.txt |

### 13.4.2 Terminal Output

When the workflow is initiated, the system parses the protocol and begins the first stage:

```
Protocol mode activated
Creating new protocol cache: protocol_425319.pkl (for formic.asc)
Resuming workflow

Workflow: Annealing → Calculation → COSMIC → Optimization → COSMIC

------------------------------------------------------------
[1/5] Annealing
------------------------------------------------------------
Using recommended box size: 8.0 Å (10.0% effective packing)
Creating 3 replicated runs in 'annealing/'...
Created: formic_1/formic_1.asc
Created: formic_2/formic_2.asc
Created: formic_3/formic_3.asc
Running 3 annealing simulation(s)

formic_1...
```

## 13.5 Execution Control and Resumption

The workflow allows for granular control over the execution pipeline via delimiters and caching.

### 13.5.1 Stage Delimiters

- **Continuous Execution (,):** A comma at the end of a protocol line indicates that the script should proceed immediately to the next stage upon completion.
- **Breakpoints (.):** A period (or dot) acts as a breakpoint. The workflow will complete the current stage and then pause, allowing the user to manually inspect intermediate files (e.g., to review **Critical** structures).

### 13.5.2 Resuming the Workflow

The system maintains a cache file (e.g., `protocol_425319.pkl`) to track progress. If a workflow is paused via a breakpoint or interrupted, it can be resumed from the last successful stage using the same command:

```
ascec formic.asc protocol
```

This ensures that expensive calculations are not repeated unnecessarily.

**Resuming the Pipeline**
Users can direct the workflow to start from a specific stage index:

`1.Annealing` → `2.Calculation` → `3.COSMIC` → `4.Optimization` → `5.COSMIC2`

**Force Restart:** To start over from the beginning of the Calculation stage (2):

```
ascec formic.asc protocol 2
```

**Resume Incomplete:** To resume the Calculation stage (2) preserving valid progress:

```
ascec formic.asc protocol 2 -i
```

## 13.6 Failure Recovery Mechanisms

The workflow incorporates robust fault-tolerance mechanisms to handle the complex potential energy surfaces often encountered in difficult systems.

### 13.6.1 Retry Logic (Execution Reliability)

This protocol addresses execution failures where a calculation does not achieve "Normal Termination" (e.g., non-zero exit codes or crashes). The script attempts to recover using one of two strategies: restarting the calculation from scratch or resuming from the last recorded coordinates. This feature is designed to mitigate transient execution errors, including filesystem disruptions and operational noise. It cannot correct systematic errors arising from incorrect input syntax; therefore, the ORCA input templates must be pre-validated for the system under study.

```
------------------------------------------------------------
[2/5] Calculation
------------------------------------------------------------
Executing calculations...

Running: opt_conf_1.inp... ✓ (2nd Attempt)
Running: opt_conf_2.inp... ✓
...
Running: opt_conf_495.inp... ✓
Running: opt_conf_496.inp... ✓

Status: 496/496 calculations completed
All calculations completed successfully
```

### 13.6.2 Redo Logic (Convergence and Topology)

The final ensemble must consist exclusively of **true local minima**. To ensure no optimized structure retains imaginary frequencies, the Redo logic automatically intervenes when topological instabilities are detected:

1. **Multiple Imaginary Frequencies:** Treated as numerical convergence artifacts. The system attempts to resolve these by re-submitting the optimization using the final coordinates of the previous run.
2. **Single Imaginary Frequency:** Treated as first-order saddle points (Transition States) or soft vibrational modes. The algorithm **perturbs the geometry along the imaginary vibrational mode** to push the structure toward a stable basin before re-optimizing.

**Computational Cost Warning:** Since this process involves high-level refinement, it is computationally expensive. It is intended to resolve soft modes and saddle points, not to compensate for an inappropriate method.

```
------------------------------------------------------------
[4/5] Optimization
------------------------------------------------------------
Stage redo enabled: max 10 attempts, target skipped ⩽ 0.0%

Executing optimization calculations...
Running: motif_01_opt.inp... ✓
...
Running: motif_35_opt.inp... ✓

Status: 35/35 optimizations completed
Optimization calculations completed

------------------------------------------------------------
[5/5] COSMIC
------------------------------------------------------------
Using cosmic script: cosmic-v01.py

Working directory: cosmic_2

Processed 2 structures with imaginary frequencies:
- 0 clustered with true minima (can be ignored)
- 2 may represent missing motifs (need recalculation)
  motif_25_opt.out  -  1 imaginary freq(s)
  motif_12_opt.out  -  2 imaginary freq(s)

Finished processing folder: orca_out_35
All selected molecular analyses complete!

Results: 5.7% critical, 5.7% skipped
→ Threshold exceeded (skipped 5.7% > 0.0%)
```

***Clarification of Metrics:*** The **Critical** set is a specific subset of the **Skipped** set. While all structures containing imaginary frequencies are automatically classified as **Skipped**, the **Critical** designation is reserved exclusively for those that form **isolated motifs**.

```
------------------------------------------------------------
Redo attempt 2/10

Processing redo structures from: cosmic_2/skipped_structures
Found 2 structure(s) to retry

motif_12_opt: 2 imaginary freqs, using final geometry ✓
motif_25_opt: 1 imaginary freq, displacing along mode ✓

Regenerated 2 input file(s) with new geometries
Resuming: Using existing optimization directory (Redo Mode)

Executing optimization calculations...
Resuming: 33/35 optimizations already completed
Running: motif_12_opt.inp... ✓
Running: motif_25_opt.inp... ✓

Status: 35/35 optimizations completed

Using cosmic script: cosmic-v01.py
python /.../cosmic-v01.py --th=2 -j4 --update-cache /tmp/tmpktk17rmd.txt
```

# 14 Results and Validation

This section presents validation case studies demonstrating the efficacy and robustness of the COSMIC ASCEC protocols. We evaluate the software across three distinct dimensions:

1. **Standalone Execution:** A hierarchical Two-Step protocol using the Water Hexamer $(H_2O)_6$ [10].
2. **Automated Workflow:** A demonstration of the fully automated pipeline utilizing the *cis*-Formic Acid Dimer [15].
3. **Clustering Algorithms:** An assessment of standard topological clustering versus RMSD-refined hierarchical clustering.

## 14.1 Standalone Execution: Water Hexamer

To validate the modular utility of the software, the Water Hexamer system (`w6.asc`) was analyzed using a manual execution strategy. This case study benchmarks the trade-off between computational cost and configurational space coverage.

### 14.1.1 Hierarchical Two-Step Protocol (ORCA)

This protocol employed a cost-effective pre-optimization filter using the semiempirical GFN2-xTB method, followed by high-level DFT refinement. The workflow processed an initial stochastic pool of **416 raw annealing candidates**.

**1. Geometric Reduction (GFN2-xTB):** The COSMIC module reduced the raw input of 416 structures to **53 representative configurations** based on topological distinctness. Figure 26 illustrates the ensemble members of a specific cluster (`cluster_1_24`), while Figure 27 displays its representative structure.

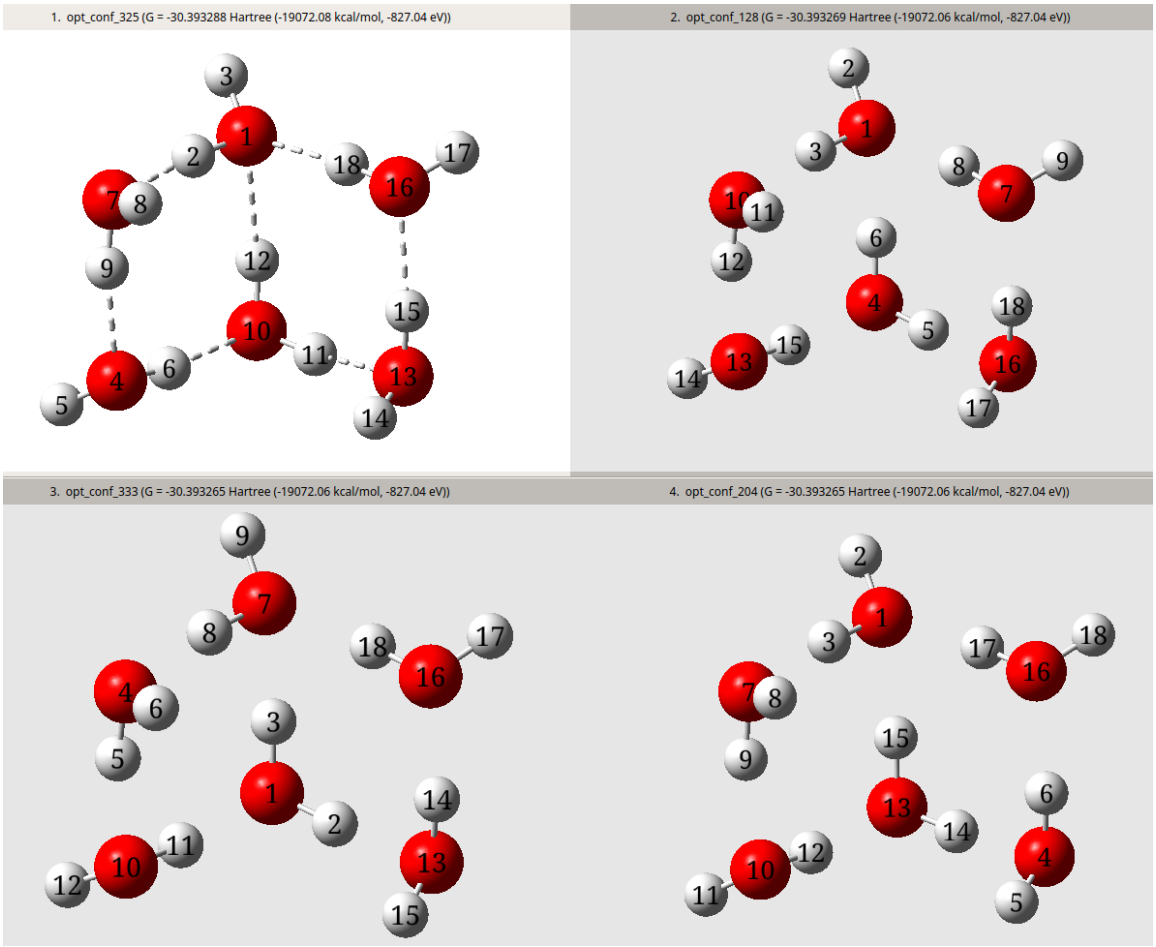



Figure 26: Visual inspection of the `cluster_1_24` ensemble members.

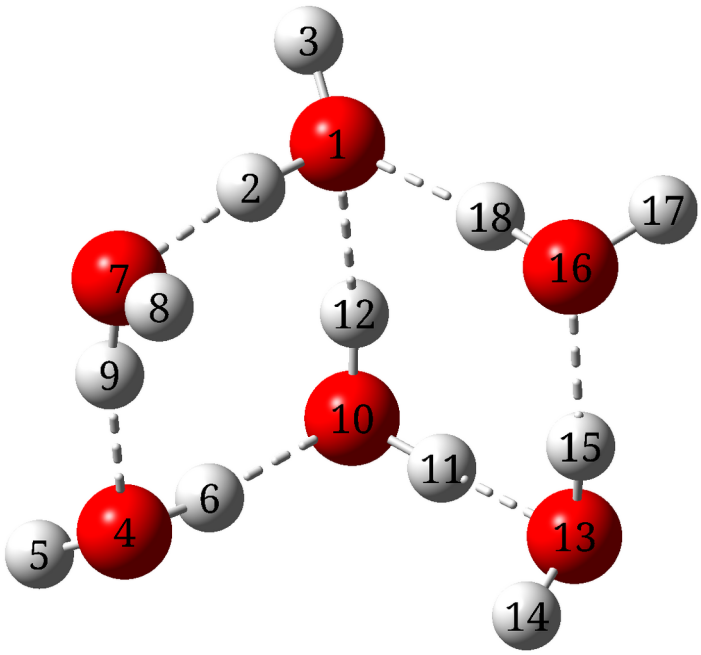


Figure 27: Representative configuration for `cluster_1_24`.

**2. High-Level Refinement (B97-3c):** The 53 representatives underwent full DFT optimization (B97-3c). A final COSMIC check (Threshold $\tau = 2.0$) consolidated these into **20 Unique Motifs**. Some of them can be seen in Figure 28. In addition, Figure 29 highlights the three most statistically significant structures.

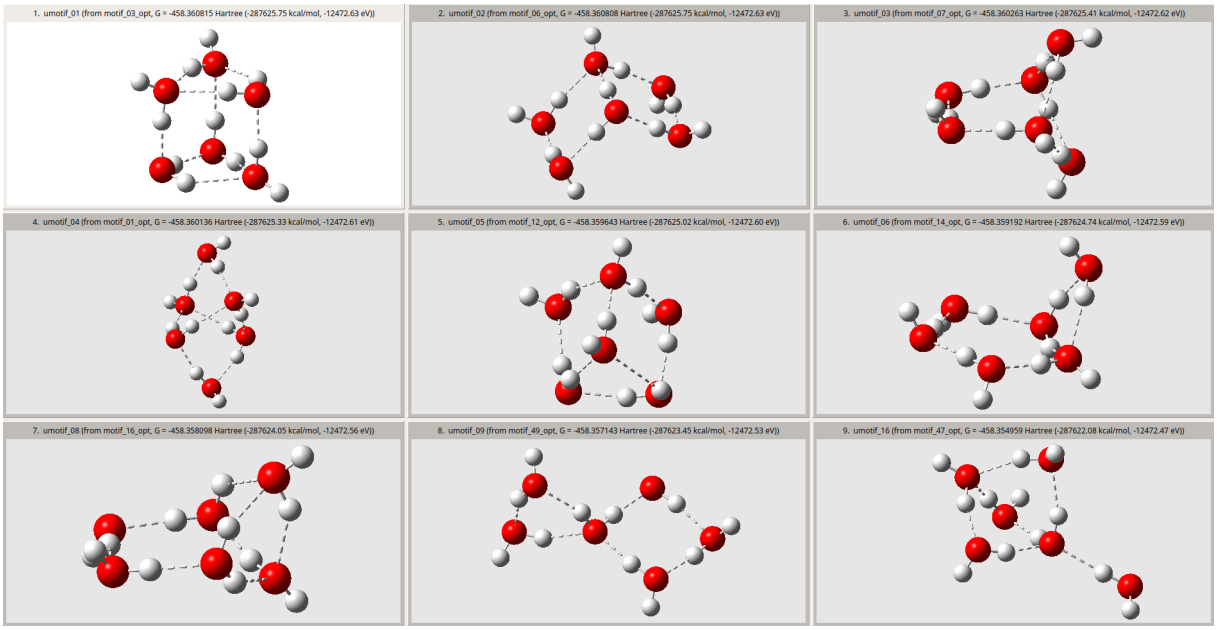



Figure 28: Subset of unique motifs identified for $(H_2O)_6$ system via the Two-Step protocol.

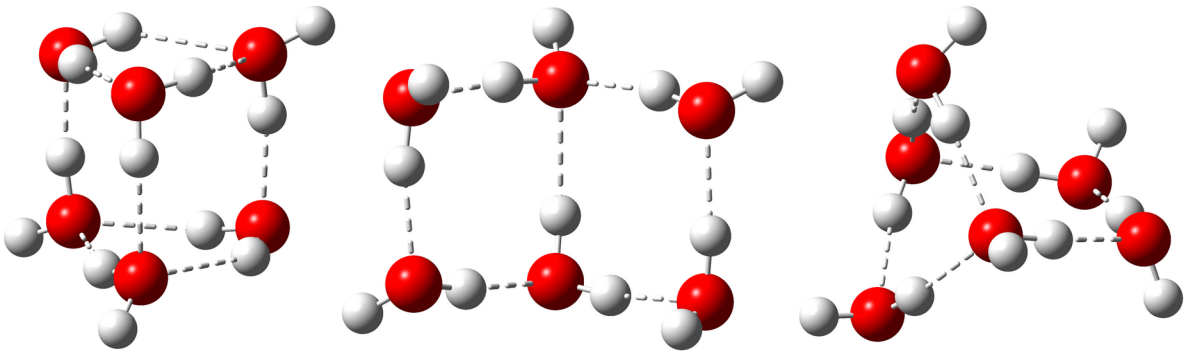


Figure 29: The three dominant water hexamer motifs identified by ASCEC.

***Boltzmann Distribution:*** Thermodynamic analysis at 298.15 K indicates that the top three motifs account for over **68%** of the total ensemble population. The global minimum (Prism-like, UMotif 01) and the first local minimum (Book-like, UMotif 02) are effectively iso-energetic.

```
Unique Motif (UMotif) Assignment Summary
(Sorted by Boltzmann population - 298.15 K)

UMotif_01 (cluster_15)
Gibbs Energy: -458.360815 Hartree (-287625.75 kcal/mol)
Population:   26.35 %
```

```
UMotif_02 (cluster_1)
Gibbs Energy: -458.360808 Hartree (-287625.75 kcal/mol)
Population:   26.14 %

UMotif_03 (cluster_2)
Gibbs Energy: -458.360263 Hartree (-287625.41 kcal/mol)
Population:   14.69 %
```

### 14.1.2 Direct Approach (Gaussian 09) & Comparative Analysis

The Gaussian 09 validation employed a direct optimization strategy at the wB97XD/6-31+G(d) level of theory, skipping the semiempirical pre-screening. Comparing the Hierarchical and Direct method approaches reveals critical insights regarding Potential Energy Surface (PES) sampling and computational efficiency.

**1. Topological Accuracy and Semiempirical Bias** Both methods successfully identified the dominant low-energy isomers (Prism, Cage, Book). However, a detailed inspection of the full ensembles reveals that the semi-empirical pretreatment (GFN2-xTB) exhibits a specific topological bias that must be taken into account.

GFN2-xTB tends to over-stabilize cooperative hydrogen bonding networks, prioritizing structures where water molecules act as both donors and acceptors. Consequently, configurations containing "pure acceptor" water molecules (which do not donate hydrogen) are occasionally filtered out during the pre-optimization step. Figure 30 illustrates such a structure, which was found by the direct approach but excluded by the two-step protocol.

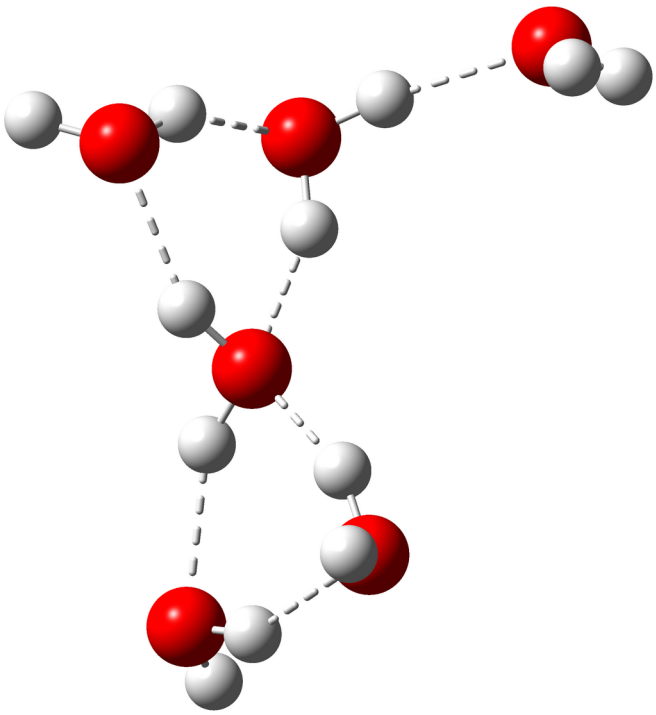


Figure 30: A specific water hexamer configuration identified only by the Direct Approach (Gaussian 09), characterized by a pure-acceptor water molecule.

Despite this topological divergence, the thermodynamic impact is negligible. The missing structure possess high relative energies and contribute $\approx 0\%$ to the Boltzmann distribution at 298.15 K. Furthermore, comparisons with high-level benchmarks (Figure 31) confirm that the ASCEC Two-Step protocol structures closely replicate the MP2/6-311++G** geometries reported by [10], in cases where the presence of pure-acceptor water molecules is **not mandatory**.

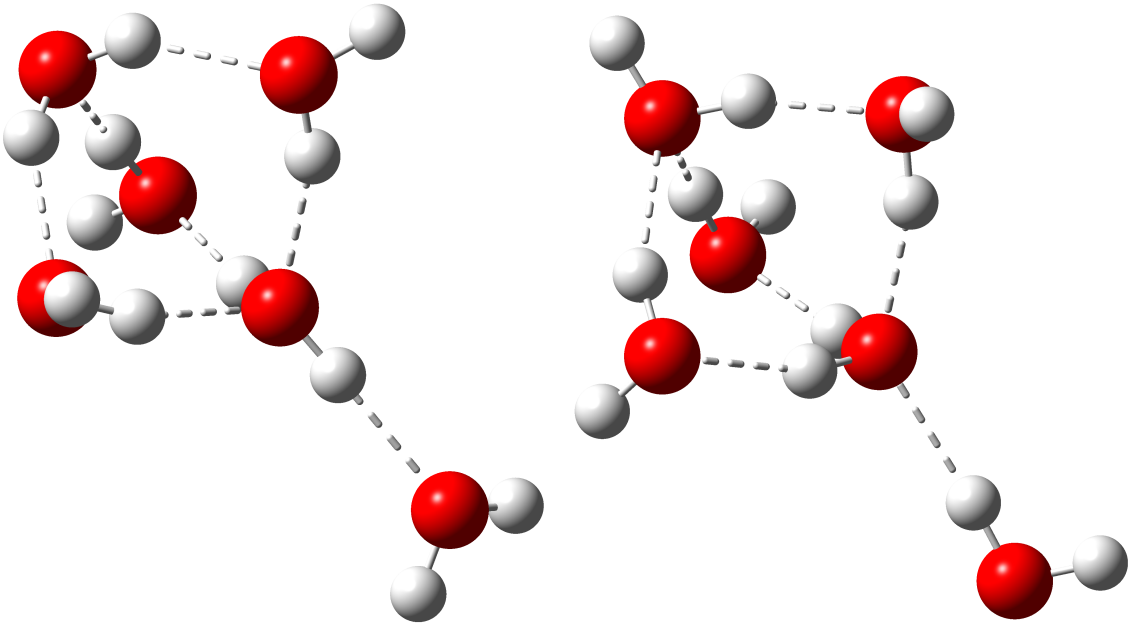


Figure 31: Structural comparison of the MP2/6-311++G** reference geometry (Left) [10] and the ASCEC-derived GFN2-xTB//B97-3c structure (Right).

**2. Computational Efficiency** The decisive advantage

## 14.2 Automated Workflow: Formic Acid Dimer

To demonstrate the autonomy of the ASCEC pipeline, the Formic Acid Dimer system was processed using the **Workflow Mode**. Unlike the manual execution, this protocol autonomously managed error correction, imaginary frequency elimination, and convergence criteria to locate the global minimum starting from a *cis*-monomer composition (Figure 32).

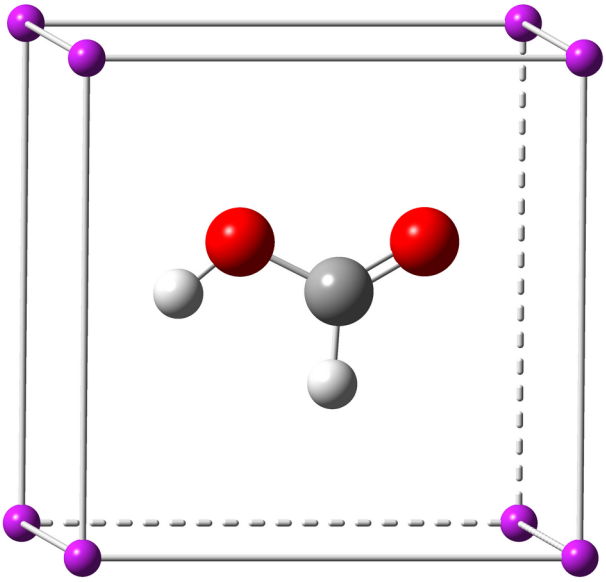


Figure 32: Initial simulation box containing *cis*-Formic Acid monomers.

### 14.2.1 Protocol and Computational Efficiency

The workflow employed a hierarchical level of theory: PM3 semiempirical potential for the annealing stage, followed by GFN2-xTB pre-optimization, and a final DFT refinement using wB97X-D3BJ/def2-TZVPP.

The entire pipeline, running on 8 cores, completed in approximately 2 days. Notably, the stochastic annealing phase accounted for **2.0%** of the total wall time, confirming that the generation of the initial candidate pool is computationally negligible compared to the quantum mechanical refinement.

Table 10: Computational timing breakdown for the automated Formic Acid workflow.

| **Stage** | **Duration** | **% Total** |
|---|---|---|
| Annealing (PM3) | 00h 53m | 2% |
| Calculation (GFN2-xTB) | 37h 41m | 83% |
| Optimization (wB97X-D3BJ) | 06h 36m | 15% |
| **Total** | **≈ 2d** | **100.0%** |

To review stage-specific completion data, refer to the protocol summary output. This file captures the full workflow sequence and provides a detailed account of error corrections applied and quality thresholds met during execution. It serves as a comprehensive audit log for both timing performance and data integrity across all stages.

```
Started:    2026-02-25 22:54:06
Completed:  2026-02-27 20:06:05
Duration:   1d 21h 11m

Workflow:   Annealing → Calc → COSMIC → Opt → COSMIC
```

```
TIMING BREAKDOWN

Stage                   Duration    % Total
--------------- --------------- ----------
Annealing           0:53:31.269       2.0%
Calculation        37:41:29.684      83.4%
Optimization        6:36:58.270      14.6%
```

### 14.2.2 Automated Error Correction and Robustness

The workflow processed an initial pool of **498 candidates**. A fundamental feature of automation is the handling of imaginary frequencies. The system detected 110 structures with imaginary modes during the preprocessing stage:

- **98 structures** were identified as redundant variations of existing minima and were safely discarded by the clustering algorithm.
- **12 structures** were flagged as potential "missing motifs" requiring vibrational displacement and re-optimization (the "Redo" loop).

**Recovery of Lost Motifs:** The "Redo" mechanism successfully recovered valid minima. For example, `opt_conf_24`, initially flagged with an imaginary frequency, was displaced along the normal mode and re-optimized, resulting in a stable unique motif (Cluster 5) that would have otherwise been lost (Figure 33).

```
Processed 110 structures with imaginary frequencies:
- 98 clustered with true minima (can be ignored)
  opt_conf_2.out  -  1 imaginary freq(s)
  ...
- 12 may represent missing motifs (need recalculation)
  opt_conf_24.out  -  1 imaginary freq(s)
  opt_conf_82.out  -  1 imaginary freq(s)
  ...
Creating 32 motifs from cluster representatives...

Results: 2.2% critical, 20.0% skipped
Threshold exceeded (critical 2.2% > 0.0%)

------------------------------------------------------------
Redo attempt 2/10

Processing redo structures from: cosmic/skipped_structures/need_recalculation
Found 12 structure(s) to retry
```

```
opt_conf_24: 1 imaginary freq, displacing along mode ✓
opt_conf_82: 1 imaginary freq, displacing along mode ✓

Creating 34 motifs from cluster representatives...

Results: 1.1% critical, 19.7% skipped
Threshold exceeded (critical 1.1% > 0.0%)
```

```
------------------------------------------------------------
```

```
Processed 108 structures with imaginary frequencies:
- 102 clustered with true minima (can be ignored)
  opt_conf_2.out  -  1 imaginary freq(s)
  ...
- 6 may represent missing motifs (need recalculation)
  opt_conf_82.out  -  1 imaginary freq(s)
  ...
```

```
Cluster 5 (1 configurations):
Files:
- opt_conf_24.out (Gibbs Energy: -22.547040 Hartree (-14148.48 kcal/mol))
```

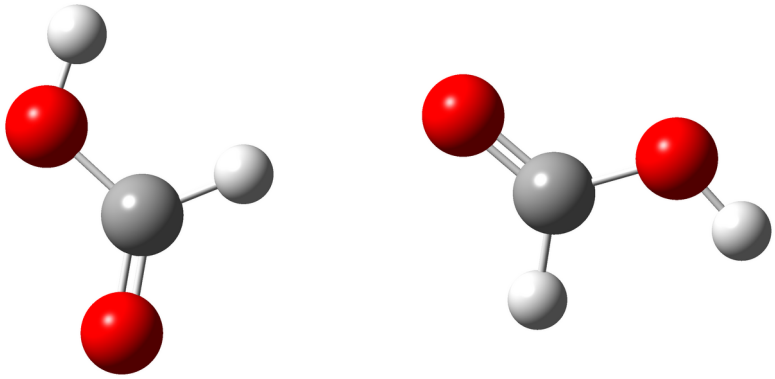


Figure 33: Configuration 24: A valid motif recovered automatically by the Redo loop mechanism.

**Residual Critical Structures:** After 10 automated refinement cycles, 6 structures remained labeled as "critical" (imaginary frequencies persisting). Manual inspection confirmed these were not algorithmic failures but rather scattering artifacts—unbroken molecules with excessive intermolecular distances ($> 7.5$ Å), as shown in Figure 34. These effectively represent dissociated states irrelevant to the bound dimer ensemble.

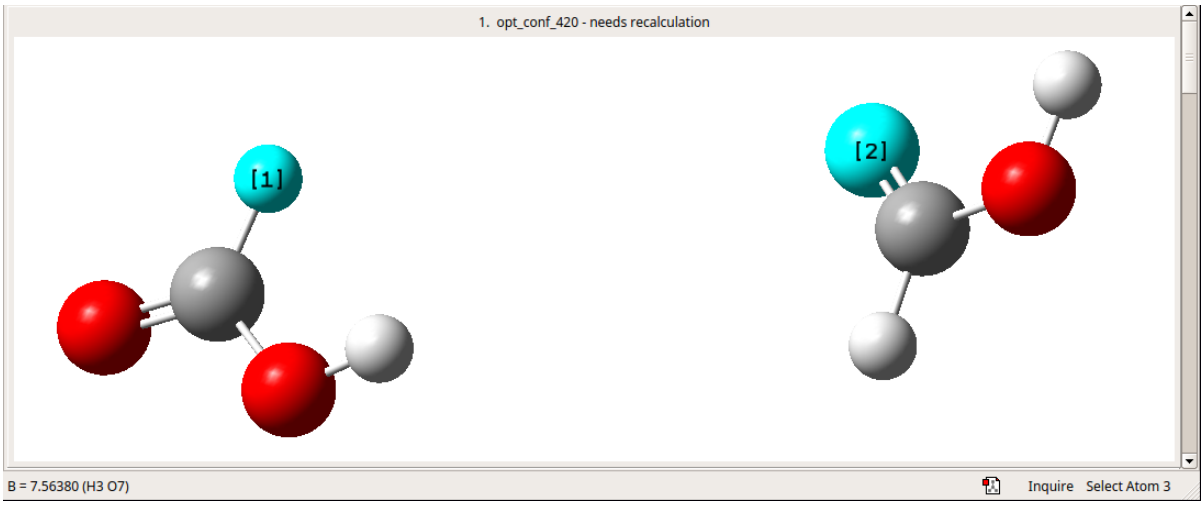



Figure 34: Representative "Critical" structure (`conf_82`) showing dissociation ($r > 7.5$ Å), correctly filtered out of the final ensemble.

### 14.2.3 Thermodynamic Analysis

The final ensemble, consolidated with a cosmic threshold of $\tau = 2.0$, yielded **12 Unique Motifs** (Figure 35). The Boltzmann distribution at 298.15 K is strongly dominated by one structure, which comprises **99.99%** of the population. This global minimum corresponds to the highly stable cyclic dimer. The next most stable motif lies $\Delta G \approx 5.45$ kcal/mol higher in energy, reflecting the penalty of breaking one of these intermolecular bonds to form an open dimer. Because this gap vastly exceeds the available ambient thermal energy ($RT \approx 0.59$ kcal/mol), all higher-energy conformers are statistically inaccessible at room temperature.

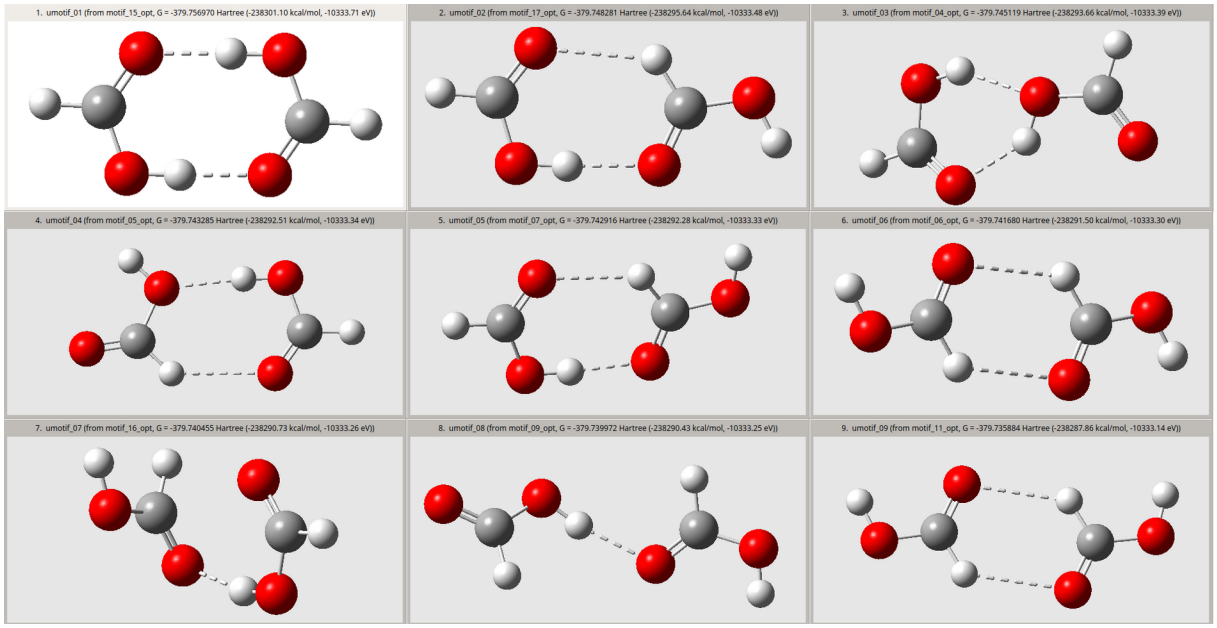



Figure 35: Subset of unique motifs identified for the formic acid system.

The global minimum (UMotif 01, Figure 36) corresponds to the cyclic dimer with a double hydrogen bond. Remarkably, although the input used *cis* monomers, the global minimum search

successfully located the energetically favorable *trans* cyclic structure, demonstrating the protocol's ability to overcome isomerization barriers.

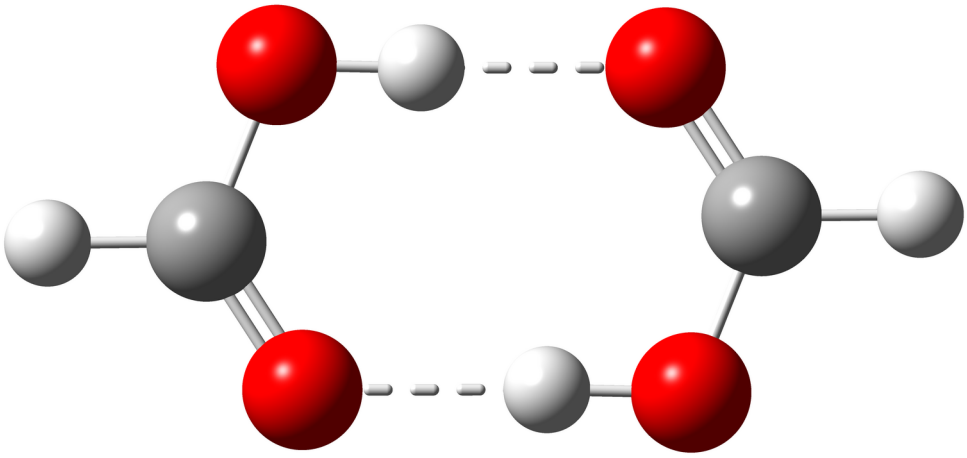


Figure 36: Global minimum: The cyclic Formic Acid Dimer.

```
Unique Motif (umotif) Assignment Summary
(sorted by Boltzmann population)

cluster_12 (umotif_01)
From structure: motif_15_opt
Gibbs Energy: -379.756970 Hartree (-238301.10 kcal/mol, -10333.71 eV)
Population: 99.99 %

cluster_10 (umotif_02)
From structure: motif_17_opt
Gibbs Energy: -379.748281 Hartree (-238295.64 kcal/mol, -10333.48 eV)
Population: 0.01 %

cluster_13 (umotif_03)
From structure: motif_04_opt
Gibbs Energy: -379.745119 Hartree (-238293.66 kcal/mol, -10333.39 eV)
Population: 0.00 %
```

## 14.3 Clustering Methodologies: Topology vs. Geometry

This section evaluates the impact of the hierarchical clustering protocols on ensemble generation. We contrast the coarse-grained Physicochemical Feature Vector (Step I) approach against an fine-grained RMSD refinement (Step II) hierarchical strategy to determine the optimal use cases for either and to decide how to choose between them.

### 14.3.1 Case Study A: Resolving Spatial Isomerism (Adenine-Thymine)

The limits of purely vector-based clustering (Step I) are evident in systems where distinct stereoisomers possess identical internal connectivity and energies. In the Adenine-Thymine dimer, the standard algorithm (Threshold $\tau = 2.0$) grouped spatially distinct configurations

into a single family (`cluster_17`). While physicochemical properties are conserved, the spatial orientation differs significantly (Figure 37).

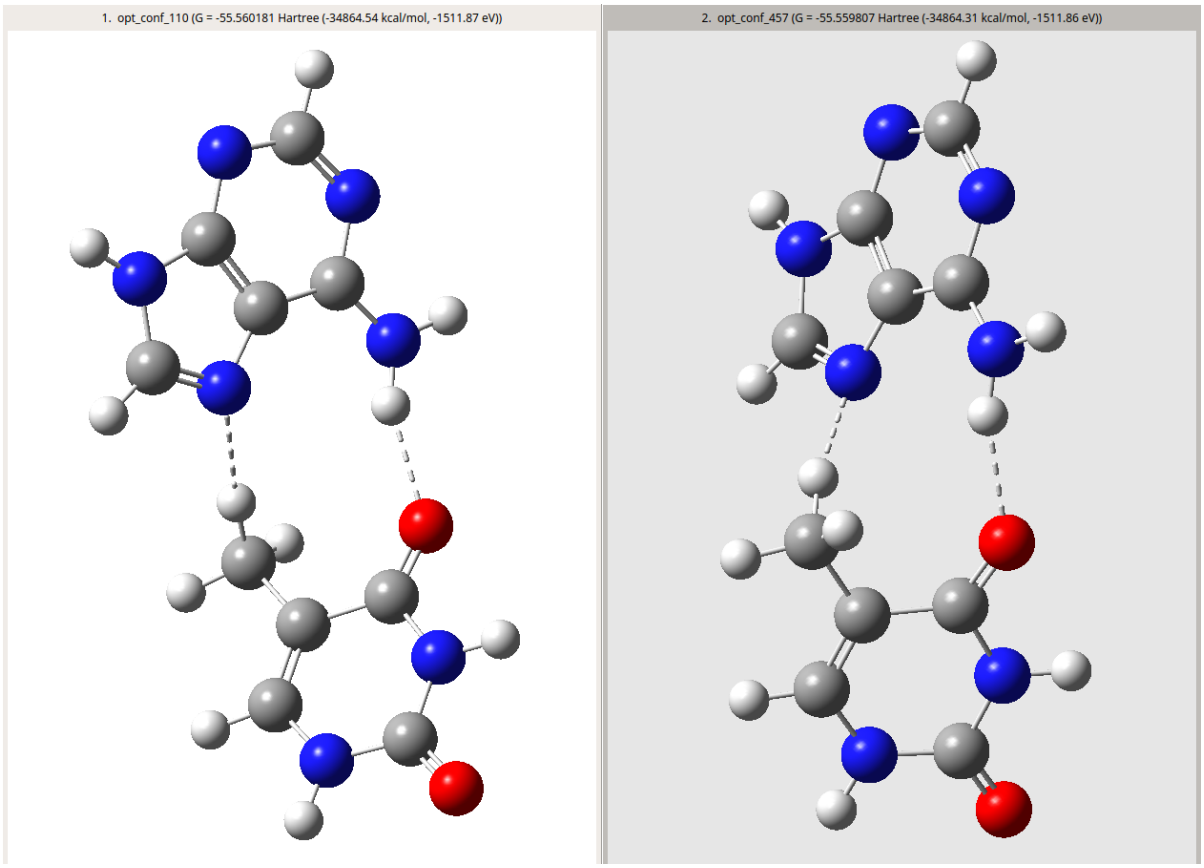



Figure 37: Two distinct spatial arrangements of the Adenine-Thymine dimer initially grouped into a single topological cluster.

By enabling the **RMSD Refinement (Step II)**, the algorithm successfully distinguished these isomers. The secondary geometric check decomposed `cluster_17` into two unique entities (`cluster_140` and `cluster_141`), despite their nearly identical Gibbs energies.

```
Cluster 140 (1 configurations) | RMSD Validated from Cluster 17:
Files:
- opt_conf_110.out (Gibbs Energy: -55.560181 Hartree (-34864.54 kcal/mol))

Cluster 141 (1 configurations) | RMSD Validated from Cluster 17:
Files:
- opt_conf_457.out (Gibbs Energy: -55.559807 Hartree (-34864.31 kcal/mol))
```

A pairwise comparison confirms the geometric distinctness, yielding a heavy-atom RMSD of $> 6.6$ Å, indicating a rotation of the complex that the scalar feature vector could not detect.

```
cosmic --compare opt_conf_110.out opt_conf_457.out
```

```
RMSD Analysis (Heavy Atoms):
Pairwise RMSD values between configurations:
opt_conf_110.out vs opt_conf_457.out: 6.655 Å
```

### 14.3.2 Case Study B: Handling Symmetry and Indexing (Arsenate Anion)

While RMSD refinement offers geometric precision, it is computationally expensive and sensitive to atom indexing, which can lead to "over-clustering" in high-symmetry systems. This is illustrated by the monohydrated dihydrogen arsenate anion ($H_2AsO_4^- \cdot H_2O$) (Figure 38).

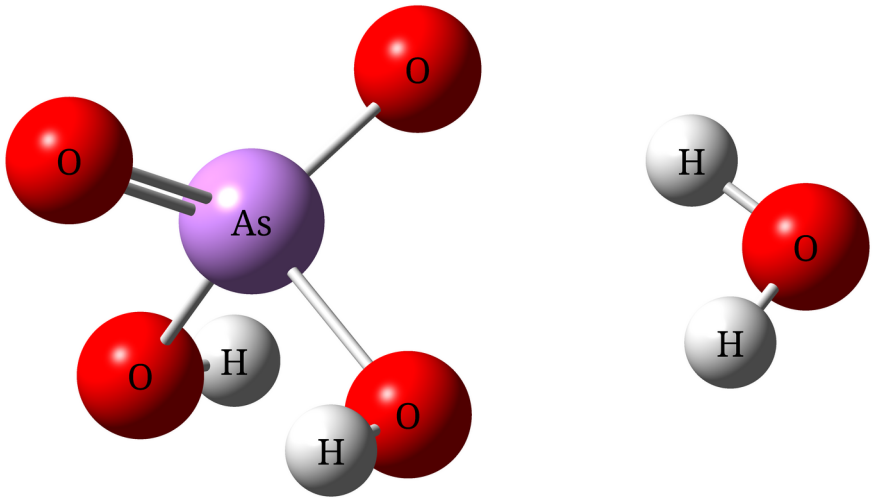



Figure 38: The monohydrated dihydrogen arsenate anion system.

The standard topological clustering (Step I) correctly identified a unified motif (`cluster_2_20`) containing 20 chemically identical structures (Figure 39).

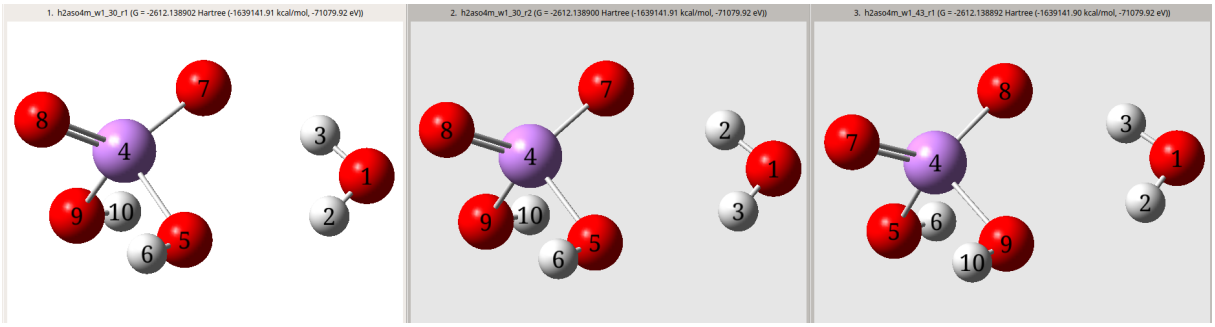


Figure 39: Correctly unified cluster identified by standard topological analysis.

However, applying RMSD refinement forced a split of this group. Due to atom indexing permutations inherent to the stochastic generation process, the RMSD algorithm flagged 8 structures as "distinct," creating a redundant artifact (`cluster_3_8`, Figure 40). These structures are chemically indistinguishable from the parent cluster.

```
Cluster 3 (8 configurations) | RMSD Validated from Cluster 2:

- h2aso4m_w1_12_r2.log (Gibbs Energy: -2612.138878 Hartree (-1639141.89 kcal/mol))
- h2aso4m_w1_38_r1.log (Gibbs Energy: -2612.138874 Hartree (-1639141.89 kcal/mol))
...
```

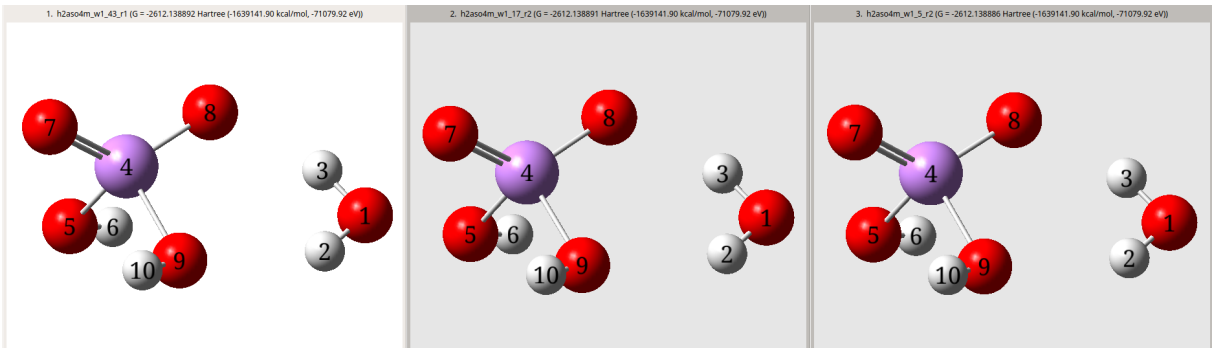



Figure 40: Redundant cluster artifact generated by RMSD sensitivity to atom indexing.

### 14.3.3 Clustering Overview

The RMSD refinement acts as a rigorous geometric filter, necessary for isolating specific stereoisomers (Case A). However, for general PES exploration, it risks inflating the ensemble size with false positives due to indexing permutations (Case B). Therefore, the ASCEC protocol defaults to topological vector clustering (Step I) for maximum efficiency, reserving RMSD (Step II) as an optional flag for systems where specific spatial orientation is critical.

# 15 Command Reference

## 15.1 Sampling and Replication

Commands used to initiate Simulated Annealing runs.

```
# Basic Execution
ascec input.asc             # Single run

# Replication (rN)
ascec input.asc r3          # 3 Replicated runs (Default packing)
ascec input.asc r5 --box10  # 5 Replicated runs with 10% packing density

# Box Validation
ascec input.asc box         # Analyzes simulation box requirements
```

## 15.2 Automated Workflow (Protocol)

Controls the execution of the multi-stage pipeline.

| Command Format | Description |
|---|---|
| `... protocol` | Starts (or resumes) the automated workflow defined. |
| `... protocol N` | **Force Restart.** Restarts the pipeline from stage $N$ (wiping progress for that stage). |
| `... protocol N -i` | **Resume Interactive.** Resumes stage $N$ from where it left off (incomplete state). |

```
Workflow: Annealing → Calculation → COSMIC → Optimization → COSMIC
```

## 15.3 Protocol thresholds

These commands generate input files for Quantum Chemistry packages (ORCA/Gaussian) and define the error handling logic.

| Flag | Description |
|---|---|
| `--retry=N` | **Execution Retries.** Maximum attempts to restart a calculation that crashed or failed (e.g., walltime limit). |
| `--redo=N` | **Topology Correction.** Maximum attempts to fix imaginary frequencies by perturbing coordinates along the unstable mode. |
| `--critical=N` | **Critical Threshold (Pre-opt).** Max % of allowed "Critical" structures (isolated imaginary freqs). Set to `0` for strict filtering. |
| `--skipped=N` | **Skipped Threshold (Final Opt).** Max % of allowed "Skipped" structures. Set to `0` to ensure all final motifs are true minima. |

## 15.4 COSMIC Analysis Module

Invoked via `ascec cosmic` or `cosmic`, this module handles clustering and Boltzmann analysis.

| Flag | Description |
|---|---|
| `--th, --threshold` | **COSMIC Threshold.** Defines the clustering granularity (Default: 1.0). Lower values = stricter clustering. |
| `--rmsd` | **Geometric Validation.** Enables RMSD checking in Ångströms (Default: 1.0 Å). Useful for distinguishing conformers with similar energies but different shapes. |
| `-j, --cores` | **Parallelization.** Number of CPU cores to use for parsing and matrix calculation (Default: Auto-detect). |
| `-T, --temperature` | **Boltzmann Temp.** Temperature in Kelvin for Gibbs Free Energy ranking (Default: 298.15 K). |
| `--weights` | **Custom Weights.** Define specific feature importance (e.g., `'(energy=0.8)(dipole=0.15)'`). |
| `--group-hb` | **H-Bond Grouping.** Groups structures by hydrogen-bond count before clustering. Produces separate dendrograms per HB family (`dendrogram_H{N}.png`). Useful for visual inspection of specific interaction networks. |
| `--reprocess-files` | Ignores the cached `.pkl` data and forces a re-read of all output. |

# 16 Quick ORCA 6.1.1 Installation for Linux

### 1. Check for AVX2 Support

To determine if your Linux system's CPU supports the AVX2 instruction set for optimal performance, execute the following command in your terminal:

```
grep -i avx2 /proc/cpuinfo
```

If this command produces any output, your CPU supports AVX2. If the output is empty, you should use the non-AVX2 version of ORCA.

### 2. Download ORCA and OpenMPI

Register and log in to the official ORCA forum to download the appropriate files for ORCA 6.1.1. You will need to download both the ORCA package and the corresponding OpenMPI version.

Compounds search: https://orcaforum.kofo.mpg.de/app.php/portal

Compounds search: https://www.open-mpi.org/

```
/
├── openmpi-4.1.8.tar.gz
├── orca_6_1_1_linux_x86-64_shared_openmpi418.tar.xz
└── orca_6_1_1_linux_x86-64_shared_openmpi418_avx2.tar.xz
```

### 3. Install OpenMPI

It is recommended to install the provided OpenMPI version before installing ORCA. Extract and compile the OpenMPI source code according to the instructions included with the download.

```
mkdir openmpi-4.1.8
```

```
tar -xvf openmpi-4.1.8.tar.gz -C openmpi-4.1.8/ --strip-components=1
```

```
cd openmpi-4.1.8
```

```
./configure --prefix=/home/user/software/openmpi-4.1.8
```

```
make -j4
```

It tells make to run 4 jobs simultaneously, you can use make -j$(nproc) instead.

```
make install
```

```
mpirun --version
mpirun (Open MPI) 4.1.8
```

### 4. Install ORCA

Extract the downloaded ORCA tarball. For example:

```
mkdir orca_6_1_1
```

```
tar -xvf orca_6_1_1_linux_x86-64_shared_openmpi418.tar.xz -C orca_6_1_1 --strip-components=1
```

This will create a directory containing the ORCA executables.

### 5. Set Environment Variables

To make the ORCA program executable from any location in your terminal, you need to add its installation directory to your system's PATH. Follow these steps:

**Open your shell's configuration file.**

Open the appropriate configuration file in a terminal-based text editor like micro. If you are using the default Bash shell, use this command:

```
micro ~/.bashrc
```

**Add the ORCA path to the file.**

*Note: The example directories for orca and mpi are located in $HOME/software.*

```
# ORCA 6.1.1 Environment Configuration

# Define the paths to your ORCA 6.1.1 installation
export ORCA_BASE=$HOME/software
export ORCA611_ROOT=$ORCA_BASE/orca_6_1_1
export OPENMPI418_ROOT=$ORCA_BASE/openmpi-4.1.8

# ORCA and OpenMPI directories to the system's PATH
export PATH="$ORCA611_ROOT:$OPENMPI418_ROOT/bin:$PATH"

# ORCA and OpenMPI libraries to the library path
export LD_LIBRARY_PATH="$ORCA611_ROOT:$OPENMPI418_ROOT/lib:$LD_LIBRARY_PATH"
```

*Save the file (Ctrl+S) and exit (Ctrl+Q). Then, reload the configuration:*

**Apply the changes.**

For the changes to take effect in your current terminal session, you must "source" the configuration file.

```
source ~/.bashrc
```

**Verify installation.**

Be careful when having the Orca reader package installed. Orca is a screen reader pre-installed on most GNOME-based Linux distributions. To remove the Orca screen reader on Linux, use the terminal command.

```
sudo apt remove --purge orca
```

```
sudo apt autoremove
```

Then verify the ORCA 6.1.1 installation

```
orca --version
```

There is no standalone version command; however, the version (e.g., Program Version 6.1.1) is displayed at the beginning of the standard output for any run.

## 17 AI Acknowledgement and Reproducibility

The development and documentation of the ASCEC protocol involved the use of large language models (LLMs) to improve code optimization, data synthesis, and technical writing. Specifically, **Claude Opus 4.5** [16] and **Gemini 3.0** [17] were used to streamline the programming process and refine the algorithmic logic.

In accordance with ethical research standards, all AI-assisted results underwent rigorous human verification to ensure scientific accuracy and reproducibility. We encourage users who adapt this methodology to maintain transparency regarding the use of computational assistants and to verify all results.

## 18 References

[1] P. J. van Laarhoven and E. H. Aarts. *Simulated Annealing: Theory and Applications*. Springer Science & Business Media, 1987.

[2] N. Metropolis et al. "Equation of State Calculations by Fast Computing Machines". In: *J. Chem. Phys.* 21.6 (1953). DOI: 10.1063/1.1699114.

[3] J. F. Pérez, C. Z. Hadad, and A. Restrepo. "Structural Studies of the Water Tetramer". In: *Int. J. Quantum Chem.* 108.10 (2008). DOI: 10.1002/qua.21615.

[4] T. M. Abramyan et al. "Cluster Analysis of Molecular Simulation Trajectories for Systems Where Both Conformation and Orientation of the Sampled States Are Important". In: *J. Comput. Chem.* 37.21 (2016). DOI: 10.1002/jcc.24416. PMID: 27292100.

[5] W. Kabsch. "A Solution for the Best Rotation to Relate Two Sets of Vectors". In: *Acta Crystallogr. Sect. A* 32.5 (1976). DOI: 10.1107/S0567739476001873.

[6] F. Neese. "Software Update: The ORCA Program System—Version 6.0". In: *WIREs Comput. Mol. Sci.* 15.2 (2025). DOI: 10.1002/wcms.70019.

[7] C. Bannwarth, S. Ehlert, and S. Grimme. "GFN2-xTB—An Accurate and Broadly Parametrized Self-Consistent Tight-Binding Quantum Chemical Method with Multipole Electrostatics and Density-Dependent Dispersion Contributions". In: *J. Chem. Theory Comput.* 15.3 (2019). DOI: 10.1021/acs.jctc.8b01176.

[8] S. Kim et al. "An Update on PUG-REST: RESTful Interface for Programmatic Access to PubChem". In: *Nucleic Acids Res.* 46.W1 (2018). DOI: 10.1093/nar/gky294.

[9] N. Rego and D. Koes. "3Dmol.Js: Molecular Visualization with WebGL". In: *Bioinformatics* 31.8 (2015). DOI: 10.1093/bioinformatics/btu829. PMID: 25505090.

[10] G. Hincapié et al. "Structural Studies of the Water Hexamer". In: *J. Phys. Chem. A* 114.29 (2010). DOI: 10.1021/jp103683m.

[11] R. R. Sokal and C. D. Michener. "A Statistical Method for Evaluating Systematic Relationships". In: *Univ. Kans. Sci. Bull.* 38.22 (1958).

[12] V. Satopaa et al. "Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior". In: *Proc. 2011 31st Int. Conf. Distrib. Comput. Syst. Workshop.* ICDCSW '11. USA: IEEE Computer Society, 2011. DOI: 10.1109/ICDCSW.2011.20.

[13] K. Pearson. "VII. Note on Regression and Inheritance in the Case of Two Parents". In: *Proc. R. Soc.* 58.347-352 (1895). DOI: 10.1098/rspl.1895.0041.

[14] R. Mojena. "Hierarchical Grouping Methods and Stopping Rules: An Evaluation*". In: *Comput J* 20.4 (1977). DOI: 10.1093/comjnl/20.4.359.

[15] P. Farfán et al. "Dimers of Formic Acid: Structures, Stability, and Double Proton Transfer". In: *J. Chem. Phys.* 147.4 (2017). DOI: 10.1063/1.4985880.

[16] Anthropic. *System Card: Claude Opus 4.5*. https://www.anthropic.com/claude-opus-4-5-system-card. 2025.

[17] Google DeepMind. *Gemini 3 Deep Think: Advancing Science, Research and Engineering*. https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-deep-think/. 2026.