profilerTools
v. 1.0

Yan M. H. Gonçalves & Bruno A. C. Horta

October 26, 2020

Contents

Chapter 1

Molecular Mechanics

All molecular-mechanics calculations are performed using custom code which depends only on standard Python libraries and `numpy`. In this chapter, we report all functional forms supported for the calculation of molecular-mechanics energies and forces. The following notation and definitions will be used for mathematical and physical quantities whenever convenient:

— $\mathbf{u} \cdot \mathbf{v}$

Dot product of vectors $\mathbf{u}$ and $\mathbf{v}$.

— $\mathbf{u} \times \mathbf{v}$

Cross product of vectors $\mathbf{u}$ and $\mathbf{v}$.

— $\mathbf{r}_i$

Position of particle i.

— $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$

Displacement from particle j to particle i.

— $r_{ij} = \sqrt{\mathbf{r}_{ij} \cdot \mathbf{r}_{ij}}$

Distance between particle i and particle j.

— $\theta_{ijk} = \arccos \frac{\mathbf{r}_{ij} \cdot \mathbf{r}_{kj}}{r_{ij} r_{kj}}$

Angle involving particles i, j and k.

— $\phi_{ijkl} = \text{sign}\left(\mathbf{r}_{ij} \cdot \mathbf{r}_{nk}\right) \arccos\left(\frac{\mathbf{r}_{mj} \cdot \mathbf{r}_{nk}}{r_{mj} r_{nk}}\right)$, with $\mathbf{r}_{mj} = \mathbf{r}_{ij} \times \mathbf{r}_{kj}$ and $\mathbf{r}_{nk} = \mathbf{r}_{kj} \times \mathbf{r}_{kl}$

Dihedral-angle involving particles i, j, k and l.

1.1 Bond stretching

The following functional forms are supported for the calculation of the energies and forces associated with a bond-stretching degree of freedom:[**?**]

— *Harmonic*

Potential energy form:

$$\begin{aligned} V(r_{ij}; k_b^h, b_0) &= \frac{1}{2}\, k_b^h (r_{ij} - b_0)^2 \quad , \\ \frac{\partial V(r_{ij}; k_b^h, b_0)}{\partial r_{ij}} &= \; k_b^h (r_{ij} - b_0) \quad , \end{aligned} \tag{1.1.1}$$

— *Quartic*

Potential energy form:

$$\begin{aligned} V(r_{ij}; k_b^q, b_0) &= \frac{1}{4}\, k_b^q (r_{ij}^2 - b_0^2)^2 \quad , \\ \frac{\partial V(r_{ij}; k_b^q, b_0)}{\partial r_{ij}} &= r_{ij} k_b^q (r_{ij}^2 - b_0^2) \quad , \end{aligned} \tag{1.1.2}$$

where $V(r_{ij})$ is the potential energy term for the bond between particles i and j.

1.2 Bond-angle stretching

The following functional forms are supported for the calculation of the energies and forces associated with a bond-angle stretching degree of freedom:[**?**, **?**]

— *Harmonic*

Potential energy form:

$$V(\theta_{ijk}; k_a^h, \theta_0) = \frac{1}{2}\ k_a^h(\theta_{ijk} - \theta_0)^2 \quad , \tag{1.2.1}$$
$$\frac{\partial V(\theta_{ijk}; k_a^h, \theta_0)}{\theta_{ijk}} = k_a^h(\theta_{ijk} - \theta_0) \quad ,$$

— *Cosine-harmonic*

Potential energy form:

$$V(\theta_{ijk}; k_a^c, \theta_0) = \frac{1}{2}\ k_a^c(\cos\theta_{ijk} - \cos\theta_0)^2 \quad , \tag{1.2.2}$$
$$\frac{\partial V(\theta_{ijk}; k_a^c, \theta_0)}{\theta_{ijk}} = -k_a^c \sin\theta_{ijk}(\cos\theta_{ijk} - \cos\theta_0) \quad ,$$

— *Urey-Bradley*

Potential energy form:

$$V(\theta_{ijk}, r_{ik}; k_a^{ub}, \theta_0^{ub}, k_{13}^{ub}, b_{13}^{ub}) = \frac{1}{2}\ k_a^{ub}(\theta_{ijk} - \theta_0)^2 + \frac{1}{2}\ k_{13}^{ub}(r_{ik} - b_{13}^{ub})^2 \quad , \tag{1.2.3}$$
$$\frac{\partial V(\theta_{ijk}, r_{ik}; k_a^{ub}, \theta_0^{ub}, k_{13}^{ub}, b_{13}^{ub})}{\partial \theta_{ijk}} = k_a^{ub}(\theta_{ijk} - \theta_0) \quad ,$$
$$\frac{\partial V(\theta_{ijk}, r_{ik}; k_a^{ub}, \theta_0^{ub}, k_{13}^{ub}, b_{13}^{ub})}{\partial r_{ik}} = k_{13}^{ub}(r_{ik} - b_{13}^{ub}) \quad ,$$

where $V(\theta_{ijk}, [r_{ik}])$ is the potential energy term for the angle involving particles i, j and k. Note that the Urey-Bradley form introduces dependence on an additional degree of freedom, the distance r_{ik}, which is treated as a harmonic bond between particles i and k.

1.3 Proper dihedral-angle torsion

The following functional forms are supported for the calculation of the energies and forces associated with a proper dihedral-angle torsion degree of freedom:[**?**, **?**]

— *Standard periodic*

Potential energy form:

$$V(\phi_{ijkl}; \{k_{d,m}^s, \phi_{0,m}\}_{m=1..6}) = \sum_{m=1}^{N_T} k_{d,m}^s\ (1 + \cos\phi_{0,m}\cos m\phi_{ijkl}) \quad , \tag{1.3.1}$$
$$\frac{\partial V(\phi_{ijkl}; \{k_{d,m}^s, \phi_{0,m}\}_{m=1..6})}{\partial \phi_{ijkl}} = -\sum_{m=1}^{N_T} m k_{d,m}^s \cos\phi_{0,m}\sin m\phi_{ijkl} \quad ,$$

with $\phi_{0,m} = 0, \pi$.

— *Ryckaert-Bellemans*

Potential energy form:

$$V(\phi_{ijkl}; \{c_m\}_{m=0..5}) = \sum_{m=0}^{5} c_m \cos^m(\phi_{ijkl} - \pi) \quad , \tag{1.3.2}$$
$$\frac{\partial V(\phi_{ijkl}; \{c_m\}_{m=0..5})}{\partial \phi_{ijkl}} = \sum_{m=1}^{5} m c_m \sin\phi_{ijkl}\cos^{m-1}(\phi_{ijkl} - \pi) \quad ,$$

— *Fourier*

Potential energy form:

$$V(\phi_{ijkl};\{f_m\}_{m=1..4}) = \frac{1}{2}\sum_{m=1}^{4} f_m\left[1+(-1)^{m+1}\cos m\phi_{ijkl}\right] \quad , \tag{1.3.3}$$
$$\frac{\partial V(\phi_{ijkl};\{f_m\}_{m=1..4})}{\partial \phi_{ijkl}} = \frac{1}{2}\sum_{m=1}^{4} m f_m(-1)^m \sin m\phi_{ijkl} \quad ,$$

Internally, this potential is computed as a Ryckaert-Bellemans potential with

$$\begin{aligned} c_0 &= f_2 + \frac{1}{2}(f_1+f_3) \\ c_1 &= \frac{1}{2}(-f_1+3f_3) \\ c_2 &= -f_2+4f_4 \\ c_3 &= -2f_3 \\ c_4 &= -4f_4 \\ c_5 &= 0 \end{aligned}$$

— *Harmonic restraint*

Potential energy form:

$$V(\phi_{ijkl};k_d^r,\phi_0) = \frac{1}{2}k_d^r(\phi_{ijkl}-\phi_0)^2 \quad , \tag{1.3.4}$$
$$\frac{\partial V(\phi_{ijkl};k_d^r,\phi_0)}{\partial \phi_{ijkl}} = k_d^r(\phi_{ijkl}-\phi_0) \quad ,$$

where $V(\phi_{ijkl})$ is the potential energy term for the proper dihedral-angle involving particles i, j, k and l.

1.4 Improper dihedral-angle bending

The following functional forms are supported for the calculation of the energies and forces associated with an improper dihedral-angle bending degree of freedom:[**?**, **?**]

— *Harmonic*

Potential energy form:

$$V(\phi_{ijkl};k_i^h,\phi_0) = \frac{1}{2}k_i^h(\phi_{ijkl}-\phi_0)^2 \quad , \tag{1.4.1}$$
$$\frac{\partial V(\phi_{ijkl};k_i^h,\phi_0)}{\phi_{ijkl}} = k_i^h(\phi_{ijkl}-\phi_0) \quad ,$$

— *Periodic*

Potential energy form:

$$V(\phi_{ijkl};k_i^p,\phi_0,m) = k_i^p\ (1+\cos\phi_0\cos m\phi_{ijkl}) \quad , \tag{1.4.2}$$
$$\frac{\partial V(\phi_{ijkl};k_i^p,\phi_0,m)}{\partial \phi_{ijkl}} = -mk_i^p\cos\phi_0\sin m\phi_{ijkl} \quad ,$$

with $\phi_0 = 0, \pi$.

where $V(\phi_{ijkl})$ is the potential energy term for the improper dihedral-angle involving particles i, j, k and l.

1.5 Nonbonded interactions

1.5.1 Standard interactions

The nonbonded interactions are calculated for every pair of non-excluded particles, without applying any type of cutoff-based truncation. The electrostatic and van der Waals interactions are calculated based on the following functional forms, respectively:[**?**, **?**]

— *Coulomb*

$$V(r_{ij};q_i,q_j) = \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}} \quad , \tag{1.5.1}$$
$$\frac{\partial V(r_{ij};q_i,q_j)}{\partial r_{ij}} = -\frac{q_i q_j}{4\pi\epsilon_0 r_{ij}^2} \quad ,$$

— *Lennard-Jones*

$$V(r_{ij};C_{6,ij},C_{12,ij}) = \frac{C_{12,ij}}{r_{ij}^{12}} - \frac{C_{6,ij}}{r_{ij}^{6}} \quad , \tag{1.5.2}$$
$$\frac{\partial V(r_{ij};C_{6,ij},C_{12,ij})}{r_{ij}} = -\frac{6}{r_{ij}^{8}}\left(\frac{2C_{12,ij}}{r_{ij}^{6}} - C_{6,ij}\right) \quad ,$$

where the interaction between particles i and j is considered.

When the pair-specific $C_{12,ij}$ and $C_{6,ij}$ Lennard-Jones parameters are not explicitly defined, they are obtained from atomic counterparts, using one of the following mixing rules:

— *Geometric*

$$C_{6,ij} = \sqrt{C_{6,i}C_{6,j}} \quad , \quad C_{12,ij} = \sqrt{C_{12,i}C_{12,j}} \tag{1.5.3}$$

— *Lorentz-Berthelot*

$$C_{6,ij} = \frac{C_{6,i}C_{6,j}}{2^6\sqrt{C_{12,i}C_{12,j}}}\left[\left(\frac{C_{12,i}}{C_{6,i}}\right)^{1/6} + \left(\frac{C_{12,j}}{C_{6,j}}\right)^{1/6}\right]^6 \tag{1.5.4}$$
$$C_{12,ij} = \frac{C_{6,i}C_{6,j}}{2^{12}\sqrt{C_{12,i}C_{12,j}}}\left[\left(\frac{C_{12,i}}{C_{6,i}}\right)^{1/6} + \left(\frac{C_{12,j}}{C_{6,j}}\right)^{1/6}\right]^{12} \quad .$$

Internally, the C_6-C_{12} representation of the Lennard-Jones potential is always used. However, some force fields are traditionally specified in terms of the σ-ϵ representation of the Lennard-Jones potential:

— *Lennard-Jones* (σ-ϵ)

$$V(r_{ij};\sigma_{ij},\epsilon_{ij}) = 4\epsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12}\right] \ , \ \sigma_{ij} = (C_{12,ij}/C_{6,ij})^{1/6} \ , \ \epsilon_{ij} = C_{6,ij}^2/4C_{12,ij} \tag{1.5.5}$$

with corresponding mixing rules

— *Geometric*

$$\sigma_{ij} = \sqrt{\sigma_i\sigma_j} \quad , \quad \epsilon_{ij} = \sqrt{\epsilon_i\epsilon_j} \tag{1.5.6}$$

— *Lorentz-Berthelot*

$$\sigma_{ij} = \frac{1}{2}(\sigma_i + \sigma_j) \quad , \quad \epsilon_{ij} = \sqrt{\epsilon_i\epsilon_j} \quad . \tag{1.5.7}$$

1.5.2 1–4 interactions

The interactions of pairs of particles that are identified as sharing a third-neighbor relationship (1–4 interactions) are treated in a different manner. First, all 1–4 Coulomb interactions are scaled down from their standard values by a common factor f_{QQ} as

$$V(r_{ij};q_i,q_j) = f_{QQ}\ \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}} \quad , \quad 0 \le f_{QQ} \le 1 \ . \tag{1.5.8}$$

For the 1–4 Lennard-Jones interactions, the options listed below are available. The first is more common and establishes a simple rule for computing 1–4 Lennard-Jones parameters based on the standard Lennard-Jones parameters. The second is more uncommon (but is used *e.g.* in GROMOS force fields) and relies on defining additional atomic Lennard-Jones parameters to use exclusively with 1–4 interactions.

— Scale down the standard atomic parameters by a common factor f_{LJ}, obtaining 1–4 atomic parameters $CS_6 = f_{LJ}C_6$ and $CS_{12} = f_{LJ}C_{12}$. After that, proceed as usual, obtaining the 1–4 pair parameters $CS_{12,ij}$ and $CS_{6,ij}$ *via* mixing rule and computing the scaled-down potential energy:

$$V(r_{ij};C_{6,ij},C_{12,ij}) = \frac{CS_{12,ij}}{r_{ij}^{12}} - \frac{CS_{6,ij}}{r_{ij}^{6}} = f_{LJ}\left(\frac{C_{12,ij}}{r_{ij}^{12}} - \frac{C_{6,ij}}{r_{ij}^{6}}\right) \quad , \quad 0 \le f_{LJ} \le 1 \tag{1.5.9}$$

— Instead of using a fixed factor f_{LJ} to modify the atomic parameters, specify an additional arbitrary set of atomic parameters CS_6 and CS_{12} which are exclusive to 1–4 interactions. The 1–4 pair parameters $CS_{12,ij}$ and $CS_{6,ij}$ are then obtained *via* mixing rule, resulting in a pair-specific scaled-down potential energy

$$V(r_{ij}; CS_{6,ij}, CS_{12,ij}) = \frac{CS_{12,ij}}{r_{ij}^{12}} - \frac{CS_{6,ij}}{r_{ij}^{6}} \tag{1.5.10}$$

1.6 Forces

The atomic forces due to the potential energy terms reported above are obtained by computing the analytical derivatives of the corresponding expressions with respect to the atomic positions involved in the degree of freedom. In general,

— *Bond terms and nonbonded terms*

$$\mathbf{f}_\alpha = -\frac{\partial V(r_{ij})}{\partial r_{ij}} \nabla_{\mathbf{r}_\alpha} r_{ij} \quad , \quad \alpha = i, j \quad , \tag{1.6.1}$$

— *Angle terms*

$$\mathbf{f}_\alpha = -\frac{\partial V(\theta_{ijk})}{\partial \theta_{ijk}} \nabla_{\mathbf{r}_\alpha} \theta_{ijk} \quad , \quad \alpha = i, j, k \quad , \tag{1.6.2}$$

— *Dihedral terms*

$$\mathbf{f}_\alpha = -\frac{\partial V(\phi_{ijkl})}{\partial \phi_{ijkl}} \nabla_{\mathbf{r}_\alpha} \phi_{ijkl} \quad , \quad \alpha = i, j, k, l \quad , \tag{1.6.3}$$

where $\mathbf{f}_\alpha$ is the force acting on particle α and $\nabla_{\mathbf{r}_\alpha}$ denotes the gradient with respect to the position of particle α. Each expression above is the product of a straightforward uni-dimensional derivative involving the potential expression with a more complicated "geometrical" derivative. The former terms are reported in the Sections above for each functional form. The latter terms are common to all interactions and are listed below:

— *Bond terms and nonbonded terms*

$$\begin{aligned} \nabla_{\mathbf{r}_i} r_{ij} &= +\frac{\mathbf{r}_{ij}}{r_{ij}} \\ \nabla_{\mathbf{r}_j} r_{ij} &= -\frac{\mathbf{r}_{ij}}{r_{ij}} \end{aligned} \tag{1.6.4}$$

— *Angle terms*

$$\begin{aligned} \nabla_{\mathbf{r}_i} \theta_{ijk} &= +\frac{1}{r_{ij}} \left(\frac{\mathbf{r}_{kj}}{r_{kj}} - \frac{\mathbf{r}_{ij}}{r_{ij}} \cos\theta_{ijk} \right) \\ \nabla_{\mathbf{r}_j} \theta_{ijk} &= -\nabla_{\mathbf{r}_i} \theta_{ijk} - \nabla_{\mathbf{r}_k} \theta_{ijk} \\ \nabla_{\mathbf{r}_k} \theta_{ijk} &= +\frac{1}{r_{kj}} \left(\frac{\mathbf{r}_{ij}}{r_{ij}} - \frac{\mathbf{r}_{kj}}{r_{kj}} \cos\theta_{ijk} \right) \end{aligned} \tag{1.6.5}$$

— *Dihedral terms*

$$\begin{aligned} \nabla_{\mathbf{r}_i} \phi_{ijkl} &= +\frac{r_{kj}}{r_{mj}^2} \mathbf{r}_{mj} \\ \nabla_{\mathbf{r}_j} \phi_{ijkl} &= +\left[\frac{\mathbf{r}_{ij} \cdot \mathbf{r}_{kj}}{r_{kj}^2} - 1 \right] \nabla_{\mathbf{r}_i} \phi_{ijkl} - \frac{\mathbf{r}_{kl} \cdot \mathbf{r}_{kj}}{r_{kj}^2} \nabla_{\mathbf{r}_l} \phi_{ijkl} \\ \nabla_{\mathbf{r}_k} \phi_{ijkl} &= -\nabla_{\mathbf{r}_i} \phi_{ijkl} - \nabla_{\mathbf{r}_j} \phi_{ijkl} - \nabla_{\mathbf{r}_l} \phi_{ijkl} \\ \nabla_{\mathbf{r}_l} \phi_{ijkl} &= +\frac{r_{kj}}{r_{nk}^2} \mathbf{r}_{nk} \quad , \end{aligned} \tag{1.6.6}$$

where

$$\mathbf{r}_{mj} = \mathbf{r}_{ij} \times \mathbf{r}_{kj} \ , \ \mathbf{r}_{nk} = \mathbf{r}_{kj} \times \mathbf{r}_{kl} \quad .$$

1.7 Choice of functional forms

The functional forms used for each interaction term are selected in the input STP file via type codes. Likewise, the Lennard-Jones potential representation, mixing rule and 1–4 details are specified in this same file. For more information, read the STP file format section on Chapter 5.

1.8 Limitations

— Periodic boundary conditions are currently not taken into account. Please make sure your systems fit entirely within the central simulation box to avoid any problems.

— The nonbonded interactions are computed explicitly regardless of the distance between the interacting particles. No other scheme for computing the nonbonded interactions is currently supported.

— Only one functional form is accepted for each type of bonded interaction. The only exception is for proper dihedral restraints, which can be defined along with any other form of proper dihedral term.

Chapter 2

Energy Minimization

In this chapter, we briefly describe the energy minimization algorithms implemented in the program, namely *steepest-descents* and *conjugate-gradients*. Both implementations are based on the descriptions supplied in the GROMOS user manual[**?**]. The following notation will be used:

— $\mathbf{r}(t_n) = (\mathbf{r}_1(t_n), \mathbf{r}_2(t_n) \dots \mathbf{r}_N(t_n))$

Configuration of the system of N particles at the n-th step of the algorithm.

— $\mathbf{f}(t_n) = (\mathbf{f}_1(t_n), \mathbf{f}_2(t_n) \dots \mathbf{f}_N(t_n))$

Configuration of the forces of the system of N particles at the n-th step of the algorithm.

2.1 Steepest-descents

In the steepest-descents energy minimization algorithm, one successively updates the configuration by moving along the direction of the gradient of the potential energy over the configurational space. The n-th step of the steepest-descents energy minimization algorithm is as follows:

1. Calculate $\mathbf{f}(t_n)$ from the configuration $\mathbf{r}(t_n)$.
2. Compute the next configuration from

$$\mathbf{r}(t_{n+1}) = \mathbf{r}(t_n) + \frac{\Delta x(t_n)}{\sqrt{\sum_{i=1}^{N} \mathbf{f}_i(t_n) \cdot \mathbf{f}_i(t_n)}} \, \mathbf{f}(t_n) \quad . \tag{2.1.1}$$

The initial value of the step size $\Delta x(t_n)$ is specified in the input variable DX0. If the energy decreases in a given step n, $\Delta x(t_{n+1}) = \min\{1.2 \cdot \Delta x(t_n),\ \mathrm{DXM}\}$, where DXM is an input variable. If the energy increases, $\Delta x(t_{n+1}) = 0.5 \cdot \Delta x(t_n)$. The algorithm is terminated if the number of steps reaches the value specified in the input variable NMAX or if the energy difference between successive steps is less than the one specified in the input variable DELE.

For more information on the input variables above, see Chapter 5.

2.2 Conjugate-gradients

In the conjugate-gradients energy minimization algorithm, one does not move strictly along the gradient of the potential energy over the configurational space. Instead, a search direction

$$\mathbf{s}(t_n) = \mathbf{f}(t_n) + \mathbf{s}(t_{n-1}) \frac{\sum_{i=1}^{N} \mathbf{f}_i(t_n) \cdot \mathbf{f}_i(t_{n-1})}{\sum_{i=1}^{N} \mathbf{f}_i(t_{n-1}) \cdot \mathbf{f}_i(t_{n-1})} \tag{2.2.1}$$

is defined for each step, with $\mathbf{s}(t_0) = \mathbf{f}(t_0)$. Let $V|_s(s; t_n) = V(\mathbf{r}(t_n) + s \cdot \mathbf{s}(t_n))$ be the restriction of the potential energy hypersurface to this search direction and

$$v(s; t_n) = -\sum_{i=1}^{N} \mathbf{f}_i(\mathbf{r}(t_n) + s \cdot \mathbf{s}(t_n)) \cdot \mathbf{s}_i(t_n) \quad , \quad \mathbf{s}(t_n) = (\mathbf{s}_1(t_n), \mathbf{s}_2(t_n) \dots \mathbf{s}_N(t_n))$$

its derivative, where $\mathbf{f}_i(\mathbf{r}(t_n) + s \cdot \mathbf{s}(t_n))$ represents the forces on particle i calculated for the configuration $\mathbf{r}(t_n) + s \cdot \mathbf{s}(t_n)$. In terms of these real functions, the n-th step of the algorithm involves two substeps:

1. Find a small interval $I(t_n) = (a(t_n), b(t_n)]$ which is guaranteed to contain a local minimum of $V|s(s;t_n)$.

   The existence of a minimum in $I(t_n)$ is ensured by any of the following pairs of conditions:

$$\begin{cases} v(a(t_n)) & < & 0 \\ V|_s(b(t_n)) & \geq & V|_s(a(t_n)) \end{cases} \quad \text{or} \quad \begin{cases} v(a(t_n)) & < & 0 \\ v(b(t_n)) & \geq & 0 \end{cases}$$

   The loop below is used to select adequate values for $a(t_n)$ and $b(t_n)$.

   Starting with $a_1(t_n) = 0$ and $b_1(t_n) = \Delta x \ (\sum_{i=1}^{N} \mathbf{s}_i(t_n) \cdot \mathbf{s}_i(t_n))^{-1/2}$, the k-th step is:

   (a) Compute the energies and forces for the configurations $\mathbf{r}(t_n) + a_k(t_n) \cdot \mathbf{s}(t_n)$ and $\mathbf{r}(t_n) + b_k(t_n) \cdot \mathbf{s}(t_n)$.

   (b) Verify the terminating conditions above. If none of them are satisfied, proceed with $a_{k+1}(t_n) = b_k(t_n)$ and $b_{k+1}(t_n) = 2b_k(t_n)$. Otherwise, $a(t_n) = a_k(t_n)$ and $b(t_n) = b_k(t_n)$.

   The step size Δx is obtained from the input variable DX0. The loop is terminated with an error if, at any step, $b_{k+1}(t_n) \geq$ DXM, where DXM is an input variable.

2. Obtain an estimate s_m of the local minimum in $I(t_n)$ based on a cubic interpolation:

$$\begin{aligned} s_m &= b - \frac{(b(t_n) - a(t_n))[W - Z + v(b(t_n))]}{v(b(t_n)) - v(a(t_n)) + 2W} \quad , \\ W &= \sqrt{Z^2 - v(a(t_n))v(b(t_n))} \quad , \\ Z &= \frac{3[V|_s(a(t_n)) - V|_s(b(t_n))]}{b(t_n) - a(t_n)} + v(a(t_n)) + v(b(t_n)) \quad . \end{aligned} \tag{2.2.2}$$

   If $V|_s(s_m) < V|_s(a(t_n))$ and $V|_s(s_m) < V|_s(b(t_n))$, accept the estimate as the local minimum and proceed with

$$\mathbf{r}(t_{n+1}) = \mathbf{r}(t_n) + s_m \cdot \mathbf{s}(t_n) \quad , \tag{2.2.3}$$

   updating the search direction with Equation 2.2.1. Otherwise, repeat the interpolation setting $b(t_n) = s_m$ (if $v(s_m) \geq 0$) or $a(t_n) = s_m$ (if $v(s_m) < 0$).

The algorithm is terminated if the number of steps reaches the value specified in the input variable NMAX or if the energy difference between successive steps is less than the value of the input variable DELE. For more information on these input variables and others mentioned above, see Chapter 5.

Chapter 3

Parameter Optimization

In this chapter, we describe the structure of the parameter optimization and how it is carried out in the context of a simple genetic algorithm (GA) method implemented in the `profilerOpt` program. It is instructive to think of the optimization in terms of three main parts:

1. The definition of the boundary conditions of the generation of trial parameters, *i.e* setting the functional form of the torsional and Lennard-Jones terms involved in the GA (Section 3.1).
2. The application of these trial parameters in the modelling of the molecular systems of interest, whereby they acquire physical meaning as force-field parameters and their fitness can be assessed and used in the GA as feedback for the evolutionary process (Section 3.2, Section 3.3, Section 3.4).
3. The generation and evolution of trial parameters, controlled by the details of the GA (*e.g.* crossover and mutation operators) and carried out in conformity with the items above (Section 3.5).

3.1 Boundary Conditions

`profilerOpt` admits a fairly flexible setup of boundary conditions for the optimized parameters. In terms of the torsional parameters, the user can select either a standard or Ryckaert-Bellemans functional form, with customized summands. It is also possible to not optimize torsional parameters at all. In terms of the Lennard-Jones parameters, the user can opt on optimizing both attractive and repulsive parameters, only one of them or none of them. For example, it is possible to carry out a `profilerOpt` run where only CS_{12} and odd-multiplicity terms of a standard proper dihedral form are taken into account. The configuration of these boundary conditions is performed in the INP input file, as described in Section 5.2.2.

3.2 From Trial Parameters to Force-field Terms

In general, each molecular system contains several torsional and Lennard-Jones degrees of freedom, and not all of them should be modelled by the optimized parameters. In fact, for each system there is a specific list of *Optimized Dihedrals* (ODs) and *Optimized Pairs* (OPs) that convey the information of the dihedral-angle terms and 1–4 nonbonded pairs involved in the parameter optimization, *i.e.* whose force-field parameters should be replaced by the GA parameters. The specification of the OPs and ODs is performed in the STP file of each system, as described in Section 5.2.1.

3.3 Torsional-scan Energy Profiles

The torsional-scan dihedral angles ϕ_k are configured in the INP file, described in Section 5.2.2. For each system, they are performed along the *Reference Dihedral* (RD) specified in the corresponding STP file, as described in Section 5.2.1. The torsional-scan energy profile $\mathcal{V}_s$ for a given system s is obtained by

1. Fixing its RD at each dihedral angle ϕ_k *via* a dihedral restraint and
   a. Performing a restrained energy minization;
   b. Computing the value of the total potential energy subtracted by the dihedral-restraint energy, referred to as $\mathcal{V}_{sk}$;
2. Subtracting its unrestrained potential energies $\mathcal{V}_{sk}$ along the torsional-scan by their minima,

$$\mathcal{V}_s = \{\mathcal{V}_{sk} - \min_k \mathcal{V}_{sk},\ k = 1 \dots N_K\} \quad , \tag{3.3.1}$$

   where N_K is the number of dihedral angles of the torsional scan.

Two settings of energy minimization can be configured for the calculation of the torsional-scan energy profiles. The first is used during the GA run, every time a torsional-scan is performed for an individual. Since thousands of torsional scans are executed in a GA run, this setting should not be very costly. The second is used only once after the GA run to refine the torsional-scan energy profiles of the optimal individual. Both settings are specified in the INP file, as described in Section 5.2.2.

3.4 Reference Data and Weights

The reference data used in the optimization (usually from QM torsional scans) will be referred to by the notation r_{sk}, where s identifies the system and k identifies the step of the torsional scan. They are supplied in single-column files, one for each system considered, as described in Section 4.2.

The relative weights w_{sk} attributed to each reference data point r_{sk} in the optimization can be customized by the user. These weights affect the optimization in the manner shown in Section 3.5.4. Full control of these values can be achieved by supplying files containing the desired weights, as described in Section 4.2. If these files are not supplied, the values of the weights are derived from the settings in the INP file, as described in Section 5.2.2. There are two options: uniform weights ($w_{sk} = 1$ for all s, k) or Boltzmann weights

$$w_{sk} = \exp\left(-r_{sk}/RT\right) \quad ,$$

where R is the universal gas constant and T is a customizable temperature. Note that it is implicitly assumed that the minimum of the reference data lies at zero for each system, *i.e.* $\min_k r_{sk} = 0$ for all s.

3.5 Genetic Algorithm

In this section, we describe the main aspects of the implementation of the genetic algorithm (GA) used in the parameter optimization. In particular, we discuss the population and individual representation (Section 3.5.2), the random initialization of parameters (Section 3.5.3), the calculation of the fitness function (Section 3.5.4) and the selection of mating individuals (Section 3.5.5) and their reproduction (Section 3.5.6), with focus on the available crossover (Section 3.5.6) and mutation (Section 3.5.6) operators.

3.5.1 General Scheme

The computational scheme for the n-th generation of the GA reads,

1. Compute the torsional-scan energy profiles of the necessary individuals (Section 3.3).

   Since this is most time-consuming task of the GA, the computation of the profiles for each individual can optionally be parallelized across several processes, as described in Section 4.2. Also, the profiles are stored in memory together with each individual, so those copied from the previous generation with no change in their parameters (*i.e.* elite individuals or parents that suffered no crossover or mutation) will not have their profiles re-calculated from the input trajectories.

2. Compute the weighted RMSD and fitness of each individual (Equation 3.5.2 and Equation 3.5.3).

3. If requested in this step,

   a. Write the parameters of each individual in the TRP output file.

   b. Write the torsional-scan energy profiles of each individual in the TRE output file.

   c. Write the average and lowest weighted RMSD in the TRR output file.

4. If elitism is set up, copy the corresponding number N_{EL} of individuals with the largest fitnesses into the population of the $(n+1)$-th generation.

5. Select the pairs of individuals for reproduction (Section 3.5.5).

6. Carry out the reproduction of each pair of parents selected above (Section 3.5.6):

   a. Test the occurrence of crossover and, if positive, apply the crossover operator.

   b. Test the occurrence of mutation and, if positive, apply the mutation operator.

   c. Carry the children into the population of the $(n+1)$-th generation.

The initial population is obtained randomly, as described in Section 3.5.3. The algorithm stops when the current generation reaches the maximal value specified in the INP file (Section 5.2.2). At this point, the parameters of the best individual are written in the IFP output file, and its torsional-scan energy profiles are refined with enhanced energy minimization settings if so requested (Section 3.3).

3.5.2 Population

In the GA population, each i-th individual can most generally be visualized as an array of floating-point values

$$I_i = (CS_6(i), CS_{12}(i), \{k_m(i)\}_{m=1\ldots6}) \quad , \quad i = 1 \ldots N_P \tag{3.5.1}$$

where CS_6, CS_{12} are Lennard-Jones parameters, $\{k_m\}_{m=1\ldots6}$ are the torsional parameters and N_P is the size of the population. If the standard dihedral representation is used, these values are the force constants $k^s_{d,m}$ for each multiplicity m in Equation 1.3.1. If the Ryckaert-Bellemans representation is used, these values are the constants c_m for each $(m-1)$-th order term in Equation 1.3.2. The values of all parameters are either obtained randomly (Section 3.5.3), or as a result of the application of the crossover (Section 3.5.6) or mutation (Section 3.5.6) operators.

At a first glance, one might be confused by the absence of the phase-shift terms $\phi^s_{0,m}$ of Equation 1.3.1 in the individual representation above (for the case of the standard dihedral representation). However, note that the potential energy function in Equation 1.3.1 satisfies

$$V(\phi_{ijkl}; \ldots, k^s_{d,m'}, 0, \ldots) = V(\phi_{ijkl}; \ldots, -k^s_{d,m'}, \pi, \ldots) + 2k^s_{d,m'} \quad ,$$

which implies that a phase-shift alteration in the m'-th term is physically equivalent to a change of sign in the corresponding force constant. In fact, the last term in the expression above is configuration-independent and does not affect the torsional potential energy curves. Therefore, as long as there is no restriction on the sign of the force constants, the optimization of the phase shifts is redundant and we can take $\phi_{0,m} = 0$ deg for all individuals.*

3.5.3 Parameter Randomization

Whenever random parameters for the individuals are requested (initialization/mutation), they are obtained from three individual random distributions: one for the torsional parameters, one for CS_6 and one for CS_{12}. The types of distributions, the ranges of acceptable values and their mean and standard deviation are configured in the INP file, as described in Section 5.2.2. Four types of distributions are available for each of them: Uniform, LogNormal, Normal and LogUniform. In the case of the torsional parameters, after a sample within the acceptable range of values is drawn from the distribution, its sign can be reversed with a fixed probability specified in the INP file, as described in Section 5.2.2. For reasons discussed above, this probability must necessarily be non-zero.

3.5.4 Fitness

The *fitness* is the quantity which reflects the chance of reproduction of a given individual of the GA population. Here, it is calculated by considering the differences between the torsional-scan energy profiles of the individual and the reference-data profiles, encoding them into a real number $\mathcal{R}$ *via* a suitable metric and finally converting this distance-like value into a fitness $\mathcal{F}$.

First, each individual I_i of the population is evaluated in terms of the weighted root-mean-square deviation

$$\mathcal{R}\left(I_i; \{w_{sk}, r_{sk}\}^s_k\right) = \sqrt{\frac{\sum_{s=1}^{N_S}\sum_{k=1}^{N_K} w_{sk}(\mathcal{V}_{sk}(I_i) - r_{sk})^2}{N_S \sum_{k=1}^{N_R} w_{sk}}} \quad , \tag{3.5.2}$$

where the w_{sk}, r_{sk} are the weights and reference data described in Section 3.4 and each $\mathcal{V}_{sk}(I_i)$ is the k-th torsional-scan of $\mathcal{V}_s(I_i)$ described in Section 3.3. This distance-like quantity is converted into a fitness value *via*

$$\mathcal{F}(I_i) = \frac{[\mathcal{R}(I_i)]^{-1}}{\sum_{j=1}^{N_P}[\mathcal{R}(I_j)]^{-1}} \quad . \tag{3.5.3}$$

The $\mathcal{R}$ and $\mathcal{F}$ quantities differ in terms of normalization, ordering and interpretability. The latter is normalized over the population and should be maximized in the optimization, but has no clear physical interpretation. The former is (at least *a priori*) not normalized and should be minimized, but has a very clear physical interpretation. For these reasons, $\mathcal{F}$ is used internally in the GA as the metric of fitness, while $\mathcal{R}$ is used in the output files.

3.5.5 Selection

The selection operator $\mathcal{S}$ is responsible for obtaining a list of N_S pairs of parents $((s_{11}, s_{12}) \ldots (s_{N_S 1}, s_{N_S 2}))$ from the current population $P = (I_1 \ldots I_{N_P})$, where

$$N_S = \begin{cases} (N_P - N_{EL})/2 & \text{if } N_P - N_{EL} \text{ is even} \\ (N_P - N_{EL} + 1)/2 & \text{otherwise} \end{cases} \tag{3.5.4}$$

*For a Ryckaert-Bellemans representation, a similar argument could be applied to motivate setting $k_1 = 0$ in Equation 3.5.1 (or, equivalently, $c_0 = 0$ in Equation 1.3.2). We do not enforce this in our code, but this behavior can be achieved by setting the input variable TORSNT[1] = 0 together with FTORS = 2 if desired.

and $s_{ij} \in P$. Repetition of individuals in this list (*i.e.* $s_{ij} = s_{(ij)'}$ for $ij \neq (ij)'$) is expected and, in fact, is the practical way by which the ones with large fitness are favoured in reproduction.

The selection operator is chosen *via* the INP file, as described in Section 5.2.2, and three options are available: roulette selection, rank selection and tournament selection. In all selection operators, $2N_S$ individuals $(s_1 \dots s_{2N_S})$, $s_i \in P$ are selected from the population and then organized naturally into the pairs $((s_{11}, s_{12}) \dots (s_{N_S 1}, s_{N_S 2}))$ with $s_{11} = s_1$, $s_{12} = s_2$, $s_{21} = s_3$ and so on. The operators differ in how each individual s_i is selected.

Roulette Selection

In the roulette operator, the probability that the j-th individual of the population is chosen at each selection s_i, $i = 1 \dots 2N_S$ is proportional to its fitness, *i.e.*

$$p_j = \frac{\mathcal{F}(I_j)}{\sum_{k=1}^{N_P} \mathcal{F}(I_k)} \quad . \tag{3.5.5}$$

Rank Selection

In the rank operator, the probability that the j-th individual of the *sorted* population is chosen at each selection s_i, $i = 1 \dots 2N_S$ is inversely proportional to its position j. More specifically, assuming that the population is sorted from the indvidual with largest fitness to the individual with smallest fitness,

$$p_j = \frac{1}{j \sum_{k=1}^{N_P} k^{-1}} \quad . \tag{3.5.6}$$

This means that the first individual (best fitness) is two times more likely to be selected than the second individual (second-best fitness), three times more likely than the third (third-best fitness), and so on.

Tournament Selection

In the tournament operator, each selection s_i, $i = 1 \dots 2N_S$ is done in two stages. First, two individuals are selected randomly from the population, with no fitness-based bias. Then, the individual with the largest fitness between the two is selected as s_i. This is equivalent to selecting an individual from the *sorted* population (as in the rank operator) with the probabiliy that the j-th individual be selected set to

$$p_j = \frac{2N_P + 1 - 2j}{N_P^2} \quad . \tag{3.5.7}$$

3.5.6 Reproduction

In the reproduction stage, the $2N_S$ parents obtained with the selection operator reproduce to generate $2N_S$ children, out of which $N_P - N_{EL}$ are carried into the population of the next generation. The children are generated by applying the crossover operator on each pair of parents and then applying the mutation operator on each child. If $N_P - N_{EL}$ is even, then $2N_S = N_P - N_{EL}$ and all children are carried into the population of the next generation. If $N_P - N_{EL}$ is odd, then $2N_S = N_P - N_{EL} + 1$, and the last child generated is discarded so as to keep the population size at N_P. The following sections discuss the crossover and mutation operators in details.

Crossover

The crossover between two parents occurs with a fixed probability specified in the INP file (Section 5.2.2). In general, the crossover operator $\mathcal{C}$ specifies a rule to obtain the parameters of two child individuals $I_{a,b}$ from a mix of the parameters of the two parents $I_{1,2}$,

$$\left\{ \begin{array}{l} I_1 = (CS_6^{(1)}, CS_{12}^{(1)}, \{k_m^{(1)}\}_{m=1\dots6}) \\ I_2 = (CS_6^{(2)}, CS_{12}^{(2)}, \{k_m^{(2)}\}_{m=1\dots6}) \end{array} \right. \overset{\mathcal{C}}{\longmapsto} \left\{ \begin{array}{l} I_a = (CS_6^{(a)}, CS_{12}^{(a)}, \{k_m^{(a)}\}_{m=1\dots6}) \\ I_b = (CS_6^{(b)}, CS_{12}^{(b)}, \{k_m^{(b)}\}_{m=1\dots6}) \end{array} \right. \tag{3.5.8}$$

where each child parameter $p^{(\alpha)}$ is obtained *via* a mixing function $f_{p\alpha}$

$$p^{(\alpha)} = f_{p\alpha}(I_1, I_2), \quad \alpha = a, b, \quad p = CS_6^{(\alpha)}, CS_{12}^{(\alpha)}, \{k_m^{(\alpha)}\}_{m=1\dots6} \ .$$

The crossover operator is chosen in the INP file (Section 5.2.2), and two options are available: the arithmetic crossover or the heuristic crossover.

In the arithmetic crossover, each child parameter is obtained as the following weighted average of the parent parameters:

$$\begin{aligned} f_{pa} &= rp^{(1)} + (1-r)p^{(2)} \\ f_{pb} &= (1-r)p^{(1)} + rp^{(2)} \quad , \end{aligned} \tag{3.5.9}$$

where $r \in [0,1]$ is a random number obtained from a uniform distribution at the start of the crossover procedure. It is important to note from the expressions above that the child parameters are bounded by the parent parameters, since

$$\min\{p^{(1)}, p^{(2)}\} \leq p^{(a)}, p^{(b)} \leq \max\{p^{(1)}, p^{(2)}\} \quad .$$

As a consequence, the range of parameter values accessible *via* this crossover operator is bounded by the range of values defined in the randomization scheme (Section 3.5.3).

In the heuristic crossover, one child is obtained as in the arithmetic crossover, but the other child is allowed to explore the parameter space in the direction of largest fitness. More specifically, assuming $\mathcal{F}(I_1) \geq \mathcal{F}(I_2)$:

$$\begin{aligned} f_{pa} &= rp^{(1)} + (1-r)p^{(2)} \\ f_{pb} &= p^{(1)} + r(p^{(1)} - p^{(2)}) \quad , \end{aligned} \tag{3.5.10}$$

where r is defined as above. In this case, child a is bounded by the parents as in the arithmetic crossover, but child b is not, since

$$\begin{aligned} p^{(2)} \leq p^{(1)} \leq p^{(b)} &\text{ if } p^{(1)} \geq p^{(2)} \\ p^{(b)} \leq p^{(1)} < p^{(2)} &\text{ if } p^{(1)} < p^{(2)} \quad . \end{aligned}$$

Mutation

The mutation of an individual occurs with the fixed probability specified in the INP file (Section 5.2.2). The mutation operator consists of three steps:

1. Choose which type of interaction is affected by the mutation. When both torsional and Lennard-Jones interactions are under optimization, either can be chosen with 50% chance. When this is not the case, the one under optimization is chosen.

2. a. If torsional interactions are selected above, choose which force constant k_m (see Equation 3.5.1) is affected by the mutation. Any optimizable term (with TORSNT[I] set to 1; see Section 5.2.2) can be chosen with equal chance, while non-optimizable terms (with TORSNT[I] set to 0) cannot be chosen.

   b. If Lennard-Jones interactions are selected above, choose whether CS_6 or CS_{12} is affected by the mutation. When both terms are under optimization, either can be chosen with 50% chance. When this is not the case, the one under optimization is chosen.

3. Replace the term selected in the previous step by a random value generated according to the randomization scheme described in Section 3.5.3.

Chapter 4

Program Usage

In this chapter, we describe the usage of the two main programs of the `profilerTools` package: `profilerGen`, used to perform torsional scans and calculate the corresponding energy profiles, and `profilerOpt`, used to optimize torsional and/or 1–4 Lennard-Jones parameters based on the reproduction of reference data on torsional-scan energy profiles. The following convention will be used. When a command-line option and its value(s) are enclosed in square brackets, this means they are not mandatory. When the value for a command-line option is enclosed in angular brackets, this means it is a file; otherwise, it is an input variable (Section 5.1).

4.1 Performing Torsional Scans: `profilerGen`

To perform a torsional scan using `profilerGen`, the user has to supply the coordinates of the system of interest, an STP file containing its topology and the specification of the reference dihedral (see Section 5.2.1), and the definition of the desired torsional-scan angles. The coordinates may be of a single configuration or of a previously obtained torsional-scan trajectory. Finally, the user can optionally control the settings of the energy minimization algorithm involved in the calculation of the energy profiles.

The usage of `profilerGen` is as follows:

```
/path/to/profilerGen.py
    -c      <COORDS>            Torsional-scan trajectory or single molecular conformation (.xyz/.gro/.g96).
    -t      <STP>               Special-topology (STP) file.
    -op     PREFIX              Prefix for output files.
    -dr     RFRST RSTEP RLST    Torsional-scan angles (deg): from RFRST to RLST with a RSTEP step.
   [-dk     RFCT]               Force constant for dihedral restraint (default = 5000 kJ/(mol.rad^2)).
   [-min    MALG]               Minimization algorithm: steepest descents (1: default) or conjugate-gradients (2).
   [-dx0    DX0]                Initial step size DX0 in minimization algorithm (default = 0.05 nm).
   [-dxm    DXM]                Maximal step size DXM in minimization algorithm (default = 0.20 nm).
   [-dele   DELE]               If |E_n+1 - E_n| <= DELE, minimization stops (default = 1.0e-09).
   [-nsteps NMAX]               Maximal number of steps in minimization (default = 50000).
```

The program will write the following files to disk:

— `PREFIX.dat`: The requested torsional-scan energy profile;

— `PREFIX.xyz`: Torsional-scan trajectory;

— `PREFIX_full.dat`: Torsional-scan energy profile split into components* (Xmgrace-compatible);

— `PREFIX_1.dat...PREFIX_K.dat`: Minimization energy trajectory at each step of the torsional scan;

— `PREFIX_1.xyz...PREFIX_K.xyz`: Minimization coordinates trajectory at each step of the torsional scan,

where K is the length of the torsional scan.

For more details on the calculation of the energy profiles, see Section 3.3 and Chapter 2.

4.2 Optimizing Parameters: `profilerOpt`

To perform parameter optimization using `profilerOpt`, the user has to supply starting torsional-scan trajectories of the systems of interest, the STP files containing their topologies and specification of the reference dihedrals and optimized terms (see Section 5.2.1), the reference data to use in the optimization and the input parameters file INP (Section 5.2.2).

The usage of `profilerOpt` is as follows:

*Components: bonds, angles, proper dihedrals, improper dihedrals, proper dihedral restraints, Lennard-Jones, Coulomb. The total potential energies are not shifted to zero.

```
/path/to/profilerOpt.py
    -np NPROCS                    Number of subprocesses spawned in parallelization.
    -r  <REF_1> ... <REF_S>       Reference data files.
    -c  <COORDS_1> ... <COORDS_S> Torsional-scan trajectory files (.g96/.xyz/.gro).
    -t  <STP_1> ... <STP_S>       Special-topology files (.stp).
    -i  <INP>                     Input file containing profilerOpt parameters (.inp).
    -op PREFIX                    Prefix for output files.
   [-w  <WEI_1> ... <WEI_S>]      Weight files (default = derive weights from <INP> file).
```

where S is the number of systems. The reference (mandatory) and weight (optional) files of each system contain a single column of values where the k-th row corresponds to the k-th step of the torsional scan.

The program will write the following files to disk:

— `PREFIX.ifp`: Parameters and weighted RMSD of the optimal individual;

— `PREFIX_minim_1.dat...PREFIX_minim_S.dat`:

Enhanced energy profile of the optimal individual for each system;

— `PREFIX_minim_1.xyz...PREFIX_minim_S.xyz`:

Enhanced torsional-scan trajectory of the optimal individual for each system;

— `PREFIX_1.dat...PREFIX_S.dat`: Energy profile of the optimal individual for each system;

— `PREFIX_1.xyz...PREFIX_S.xyz`: Torsional-scan trajectory of the optimal individual for each system;

— `PREFIX_1.tre...PREFIX_S.tre`: Trajectory of individual energy profiles over generations for each system;

— `PREFIX.trp`: Trajectory of individual parameters over generations;

— `PREFIX.trr`: Trajectory of weighted RMSD values over generations,

where S is the number of systems.

The enhanced quantities mentioned above are obtained based on refined energy minimization settings, as described in Section 3.3. The IFP, TRP, TRE and TRR output file formats are described in Section 5.2.

Chapter 5

Input Variables and File Formats

The execution of the `profilerOpt` and `profilerGen` programs depends on several input variables and input data, supplied either as command-line options or in appropriate input files. Likewise, during execution, the output of these programs is written to the standard streams and to specific output files. In this chapter, we discuss the configuration of these input variables and the format of these input and output files.

5.1 Input Variables

Input variables control the details of execution of the programs and are of two major types: numeric values (integer or float) and arrays of numeric values. They can be supplied either as command-line options or in an appropriate input file, depending on the usage. For example, the input variable NPROCS (number of processes for parallelization) is an integer specified in the command-line option `-np` of `profilerOpt`; the input variable LJNT (controls the optimization of Lennard-Jones terms) is an array of two integers specified in the INP input file. A complete list of input variables and their types is given in Table 5.1 for reference.

Table 5.1: Input variables, their types, where they are set and links to more detailed description of their allowed values.

Variable	Type	Description	Set in	Block/Option	Cross-reference
`profilerOpt`					
NPROCS	Integer	Number of subprocesses spawned in parallelization.	Cmd-line	`-np`	Section 4.2
NTORS	Integer	Controls optimization of torsional parameters.	INP file	`parameter_optimization`	p. 23
NLJ	Integer	Controls optimization of Lennard-Jones parameters.	INP file	`parameter_optimization`	p. 23
FTORS	Integer	Controls functional form of torsional terms.	INP file	`parameter_optimization`	p. 23
TORSNT	Array of Integers	Fine control of functional form of torsional terms.	INP file	`parameter_optimization`	p. 23
LJNT	Array of Integers	Controls which Lennard-Jones components are optimized.	INP file	`parameter_optimization`	p. 23
WTEMP	Float	Controls weights given to reference data points.	INP file	`parameter_optimization`	p. 23
PINV	Integer	Probability (over 100) of inverting sign of random torsional parameters.	INP file	`parameter_randomization`	p. 23
DIST	Array of Integers	Controls types of distributions to draw random parameters from.	INP file	`parameter_randomization`	p. 23
MIN	Array of Float	Controls minimum values for random parameters.	INP file	`parameter_randomization`	p. 24
MAX	Array of Float	Controls maximum values for random parameters.	INP file	`parameter_randomization`	p. 24
MEAN	Array of Float	Controls mean values of the distributions to draw random parameters from.	INP file	`parameter_randomization`	p. 24
STDDEV	Array of Float	Controls std. dev. of the distributions to draw random parameters from.	INP file	`parameter_randomization`	p. 24
SEED	Integer	Sets seed of random number generator.	INP file	`seed`	p. 24
POPSIZE	Integer	Population size in GA.	INP file	`genetic_algorithm`	p. 24
NGENS	Integer	Number of generations in the GA.	INP file	`genetic_algorithm`	p. 24
SELTYPE	Integer	Selection operator in GA.	INP file	`genetic_algorithm`	p. 24
NTEL	Integer	Number of elite individuals in GA.	INP file	`genetic_algorithm`	p. 25
CRTYPE	Integer	Crossover operator in GA.	INP file	`genetic_algorithm`	p. 25
CRRATE	Integer	Probability (over 100) of crossover.	INP file	`genetic_algorithm`	p. 25
MUTRATE	Integer	Probability (over 100) of mutation.	INP file	`genetic_algorithm`	p. 25
NSTE	Integer	Frequency of writing population energies to TRE file.	INP file	`writetraj`	p. 25
NSTP	Integer	Frequency of writing population parameters to TRP file.	INP file	`writetraj`	p. 25
RFRST	Float	Starting angle of torsional scan.	INP file	`torsional_scan`	p. 25
RSTEP	Float	Step angle of torsional scan.	INP file	`torsional_scan`	p. 25
RLST	Float	Last angle of torsional scan.	INP file	`torsional_scan`	p. 25
RFCT	Float	Force constant of dihedral restraint for torsional scan.	INP file	`torsional_scan`	p. 25
MALG	Array of Integers	Minimization algorithms.	INP file	`minimization`	p. ??
DX0	Array of Floats	Initial step sizes DX0 in minimization algorithms.	INP file	`minimization`	p. 26
DXM	Array of Floats	Maximal step sizes DXM in minimization algorithms.	INP file	`minimization`	p. 26
NMAX	Array of Integers	Maximum number of steps in minimization algorithms.	INP file	`minimization`	p. 26
DELE	Array of Floats	Energy-convergence criteria of minimization algorithms.	INP file	`minimization`	p. 26
`profilerGen`					
RFRST	Float	Starting angle of torsional scan.	Cmd-line	`-dr`	Section 4.1

Table 5.1: Input variables, their types, where they are set and links to more detailed description of their allowed values.

Variable	Type	Description	Set in	Block/Option	Cross-reference
RSTEP	Float	Step angle of torsional scan.	Cmd-line	`-dr`	Section 4.1
RLST	Float	Last angle of torsional scan.	Cmd-line	`-dr`	Section 4.1
RFCT	Float	Force constant of dihedral restraint for torsional scan.	Cmd-line	`-dk`	Section 4.1
MALG	Integer	Minimization algorithm.	Cmd-line	`-min`	Section 4.1
DX0	Float	Initial step size DX0 in minimization algorithms.	Cmd-line	`-dx0`	Section 4.1
DXM	Float	Maximal step size DXM in minimization algorithms.	Cmd-line	`-dxm`	Section 4.1
NMAX	Integer	Maximum number of steps in minimization algorithm.	Cmd-line	`-dele`	Section 4.1
DELE	Float	Energy-convergence criterium of minimization algorithm.	Cmd-line	`-nsteps`	Section 4.1

Table 5.2: Representation of the Lennard-Jones parameters expected in the STP file along with the corresponding mixing rule for each allowed value of `comb-rule` in the `[ defaults ]` block.

`comb-rule`	Representation	Mixing Rule
1	C_6-C_{12}	Geometric
2	σ-ϵ	Lorentz-Berthelot
3	σ-ϵ	Geometric

5.2 File Formats

5.2.1 Special Topology File (STP)

The special topology (STP) file combines the topology with the specification of the RD (Section 3.3), the ODs and the OPs (Section 3.2). This file is organized into blocks, each starting with a line consisting of `[ blockname ]` (where `blockname` is the name of the block) and ending with an empty line or with the start of a new block. The structure of the STP file is based on three parts, composed by the following blocks:

1. Topology and interaction parameters:

   `[ defaults ]`; `[ atoms ]`; `[ nbpairs ]`; `[ bonds ]`; `[ angles ]`; `[ dihedrals ]`.

2. Reference dihedral:

   `[ refdihedral ]`.

3. Optimization targets:

   `[ optdihedrals ]`; `[ optpairs ]` or `[ optatoms ]`.

Topology and Interaction Parameters

Users who are familiar with the GROMACS package will notice that the format of the STP file looks very similar to that of the GROMACS ITP file, specially in its topology part. In the following, we discuss each of these blocks.

`[ defaults ]` The format of this block is similar to its correspondent in the ITP file format*. The following variables are specified in order:

— `nbfunc`: **(Integer)** the non-bonded function type. Only the value 1 (Lennard-Jones) is supported;

— `comb-rule`: **(Integer)** the number of the combination rule and Lennard-Jones potential representation (see Table 5.2). Note that all molecular-mechanics calculations are ultimately performed in the C_6-C_{12} representation, regardless of the representation expected in the STP file.

— `gen-pairs`: **(String)** the strategy of automatic obtention of 1–4 Lennard-Jones pair parameters based on the mixing rule. The allowed values are:

  — `no`: do not generate pair parameters. This requires explicitly setting the interaction parameters for all 1–4 nonbonded pairs in the `[ nbpairs ]` block;

  — `fudge`: compute the pair parameters by applying the mixing rule on the standard atomic parameters (C_6, C_{12}) and then scaling the results by `fudgeLJ` (f_{LJ});

  — `atomic`: compute the pair parameters by applying the mixing rule on the 1–4 atomic parameters (CS_6, CS_{12}). This requires setting CS_6 and CS_{12} for all atoms in the `[ atoms ]` block.

— `fudgeLJ`: **(Float)** scaling factor f_{LJ} used to obtain 1–4 Lennard-Jones pair parameters from the standard atomic parameters when `gen-pairs` is `fudge`.

— `fudgeQQ`: **(Float)** scaling factor f_{QQ} used to obtain 1–4 electrostatic pair parameters from the atomic partial charges.

For more details about the calculation of the pair parameters using the mixing rule and about the scaling factors f_{LJ} and f_{QQ}, see Section 1.5.

*Strictly speaking, this block is in most cases imported into the ITP file from the force-field file, so it is not directly a part of the ITP file.

Table 5.3: Interaction types, interaction type codes and list of corresponding parameters for use in the topology part of the STP file.

Interaction type	Block	Code	List of parameters	Cross-reference
Nonbonded				
Standard pair	`nbpairs`	1	C_6 (kJ mol^{-1}nm^{-6}); C_{12} (kJ mol^{-1}nm^{-12})$^{(a)}$ σ (nm); ϵ (kJ mol^{-1})$^{(b)}$	Equation 1.5.2
1–4 pair	`nbpairs`	2	CS_6 (kJ mol^{-1}nm^{-6}); CS_{12} (kJ mol^{-1}nm^{-12})$^{(a)}$ σ_S (nm); ϵ_S (kJ mol^{-1})$^{(b)}$	Equation 1.5.2
Bonds				
Harmonic	`bonds`	1	b_0 (nm); k_b^h (kJ mol^{-1}nm^{-2})	Equation 1.1.1
Quartic	`bonds`	2	b_0 (nm); k_b^q (kJ mol^{-1}nm^{-4})	Equation 1.1.2
Angles				
Harmonic	`angles`	1	θ_0 (deg); k_a^h (kJ mol^{-1}rad^{-2})	Equation 1.2.1
Cosine-harmonic	`angles`	2	θ_0 (deg); k_a^c (kJ mol^{-1})	Equation 1.2.2
Urey-Bradley	`angles`	5	θ_0 (deg); k_a^{ub} (kJ mol^{-1}rad^{-2}); b_{13}^{ub} (nm); k_{13}^{ub} (kJ mol^{-1}nm^{-2})	Equation 1.2.3
Proper dihedrals				
Standard periodic$^{(c)}$	`dihedrals`	1	$\phi_{0,m}$ (deg); $k_{d,m}^s$ (kJ mol^{-1}); m	Equation 1.3.1
Ryckaert-Bellemans	`dihedrals`	3	c_0; c_1; c_2; c_3; c_4; c_5 (kJ mol^{-1})	Equation 1.3.2
Fourier	`dihedrals`	5	f_1; f_2; f_3; f_4 (kJ mol^{-1})	Equation 1.3.3
Standard periodic$^{(d)}$	`dihedrals`	9	$\phi_{0,m}$ (deg); $k_{d,m}^s$ (kJ mol^{-1}); m	Equation 1.3.1
Improper dihedrals				
Harmonic	`dihedrals`	2	ϕ_0 (deg); k_i^h (kJ mol^{-1}rad^{-2})	Equation 1.4.1
Periodic	`dihedrals`	4	ϕ_0 (deg); k_i^p (kJ mol^{-1}); m	Equation 1.4.2

$^{(a)}$: For a C_6-C_{12} representation; see Table 5.2.
$^{(b)}$: For a σ-ϵ representation; see Table 5.2.
$^{(c)}$: Using only one summand of Equation 1.3.1.
$^{(d)}$: Using one or more summands of Equation 1.3.1.

`[ atoms ]` The format of this block depends on the `gen-pairs` strategy and on `comb-rule`. If `gen-pairs` is `atomic`, each line must contain, in sequence and separated by spaces:

— Atom type **(String)**

— Lennard-Jones parameter C_6 or σ **(Float)**; see Table 5.2.

— Lennard-Jones parameter C_{12} or ϵ **(Float)**; see Table 5.2.

— Lennard-Jones parameter CS_6 or σ_S **(Float)**; see Table 5.2.

— Lennard-Jones parameter CS_{12} or ϵ_S **(Float)**; see Table 5.2.

— Atomic partial charge q **(Float)**

For other values of `gen-pairs`, CS_6 and CS_{12} are not used and may be omitted. If they are supplied, their values are skipped when parsing the file.

`[ nbpairs ]` In this block, each line describes a nonbonded pair a_i—a_j and must contain, in sequence and separated by spaces, the indexes a_i **(Integer)** and a_j **(Integer)**, the corresponding interaction type code **(Integer)** and, in some cases, the values of the Lennard-Jones interaction parameters listed in a specific order (see Table 5.3). The interaction parameters are only required when they cannot be inferred from the atomic parameters, *i.e.* for 1–4 nonbonded pairs when `gen-pairs` is `no`. When they are not required but nonetheless supplied, they *override* the default values derived using the mixing rule. In divergence with the ITP file format, note that *all* nonbonded pairs must be listed in this block, regardless if the atoms involved are third-neighbors or not.

`[ bonds ]` In this block, each line describes a bond a_i—a_j and must contain, in sequence and separated by spaces, the indexes a_i **(Integer)** and a_j **(Integer)**, the corresponding interaction type code **(Integer)** and the values of the interaction parameters listed in a specific order (see Table 5.3). Note that the format of this block is the same as its correspondent in the ITP file format, with the caveat that the interaction parameters must always be listed explicitly.

`[ angles ]` In this block, each line describes an angle a_i—a_j—a_k and must contain, in sequence and separated by spaces, the indexes a_i **(Integer)**, a_j **(Integer)** and a_k **(Integer)**, the corresponding interaction type code **(Integer)** and the values of the interaction parameters listed in a specific order (see Table 5.3). Note that the format of this block is the same as its correspondent in the ITP file format, with the caveat that the interaction parameters must always be listed explicitly.

`[ dihedrals ]` In this block, each line describes a dihedral angle a_i—a_j—a_k—a_l and must contain, in sequence and separated by spaces, the indexes a_i **(Integer)**, a_j **(Integer)**, a_k **(Integer)** and a_l **(Integer)**, the corresponding interaction type code **(Integer)** and the values of the interaction parameters listed in a specific order (see Table 5.3). Note that the format of this block is the same as its correspondent in the ITP file format, except that: (*i*) the interaction parameters must always be listed explicitly and (*ii*) this block also accomodates dihedral restraints. However, bear in mind that the dihedral restraints specified in this manner have no connection with the dihedral restraints used to perform the torsional scans in `profilerGen` and `profilerOpt` (except for the common functional form).

Reference Dihedral

The reference dihedral along which to perform the torsional scans is set in the `[ refdihedral ]` block of the STP file. This block is formed by a single line containing the indexes of the four atoms of the desired dihedral angle. For consistency, it is required that the reference dihedral match one of the dihedral angles listed in a `[ dihedrals ]` block.

Optimization Targets

Dihedrals The dihedral-angle terms affected by the optimization are specified in the `[ optdihedrals ]` block. In this block, each line describes a dihedral angle and contains the indexes of its four atoms. For consistency, every dihedral listed in this block must match one of the dihedral angles listed in a `[ dihedrals ]` block.

1–4 Pairs The third-neighbor nonbonded pairs affected by the optimization can be specified in two mutually exclusive ways. When optimization of specific nonbonded pairs is intended, the most effective strategy is to use the `[ optpairs ]` block, wherein each line describes a 1–4 nonbonded pair and contains the indexes of its two atoms. When the molecular-mechanics calculations are performed, the GA parameters override the pair parameters of all pairs listed in this block. For consistency, it is required that every pair listed in this block must match one of the 1–4 pairs listed in the `[ nbpairs ]` block.

When optimization of the 1–4 Lennard-Jones parameters of an *atomtype* is intended instead (which only makes sense if `gen-pairs` is `atomic`, which is not the case for most force fields[†]), the `[ optatoms ]` block can be used. In this case, the indexes of the atoms of the desired atomtype are listed, separated by spaces or line breaks. When the molecular-mechanics calculations are performed, the GA parameters override the atomic parameters of all atoms listed in this block. This in turn affects (*via* mixing rule) *all* 1–4 nonbonded pair interactions involving at least one of these atoms.

5.2.2 Input Parameters File (INP)

The input parameters file (INP) contains the definition of the majority of the input variables of `profilerOpt`. Like the STP file, the INP file is organized into blocks, wherein input variables of similar nature are defined according to their position. Sucessive positions in a block are separated by spaces or by a line break, with no syntactic distinction between the two. However, the use of line breaks is recommended to separate clusters of closely related variables within each block. In the following, we go through each block in succession, giving the position, description and allowed values of the input variables. Note that a complete list of the input variables and their corresponding blocks is also given in Table 5.3.

`[ parameter_optimization ]`

Positions of the variables:

NTORS	NLJ	FTORS			
TORSNT[1]	TORSNT[2]	TORSNT[3]	TORSNT[4]	TORSNT[5]	TORSNT[6]
LJNT[1]	LJNT[2]				
WTEMP					

Details of the variables:

[†]To our knowledge, only GROMOS-type force fields are compatible with setting `gen-pairs` to `atomic`.

NTORS : **(Integer)** Controls optimization of torsional parameters.
Allowed values:
0 : do not optimize
1 : optimize

NLJ : **(Integer)** Controls optimization of Lennard-Jones parameters.
Allowed values:
0 : do not optimize
1 : optimize

FTORS : **(Integer)** Controls functional form of torsional terms.
Allowed values:
1 : Standard periodic dihedral
2 : Ryckaert-Bellemans dihedral

TORSNT[I] : **(Array of Integers)** Fine control of functional form of torsional terms.
Allowed values:
0 : set I-th term to zero
1 : optimize I-th term
Array index targets:
1 : first summand of Equation 1.3.1 or Equation 1.3.2
2 : second summand of Equation 1.3.1 or Equation 1.3.2
3 : third summand of Equation 1.3.1 or Equation 1.3.2
4 : fourth summand of Equation 1.3.1 or Equation 1.3.2
5 : fifth summand of Equation 1.3.1 or Equation 1.3.2
6 : sixth summand of Equation 1.3.1 or Equation 1.3.2

LJNT[I] : **(Array of Integers)** Controls which Lennard-Jones components are optimized.
Allowed values:
0 : use I-th value derived from STP file
1 : optimize I-th term
Array index targets:
1 : CS_6
2 : CS_{12}

WTEMP : **(Float)** Controls weights given to reference data points.
Allowed values:
0 : uniform weights
> 0 : Boltzmann weights with temperature WTEMP

`[ parameter_randomization ]`

Positions of the variables:

PINV				
DIST[1]	MIN[1]	MAX[1]	MEAN[1]	STDDEV[1]
DIST[2]	MIN[2]	MAX[2]	MEAN[2]	STDDEV[2]
DIST[3]	MIN[3]	MAX[3]	MEAN[3]	STDDEV[3]

Details of the variables:

PINV : **(Integer)** Probability (over 100) of inverting sign of random torsional parameters.
Allowed values:
$0 \ldots 100$: set probability to PINV/100

DIST[I] : **(Array of Integers)** Controls types of distributions to draw random parameters from.
Allowed values:
1 : uniform distribution
2 : LogNormal distribution
3 : normal distribution
4 : LogUniform distribution
Array index targets:
1 : torsional terms
2 : CS_6
3 : CS_{12}

MIN[I] : **(Array of Float)** Controls minimum values of random parameters.
Allowed values:
$\in \mathbb{R}$: only accept random values larger than MIN[I]
Array index targets:
1 : torsional terms
2 : CS_6
3 : CS_{12}

MAX[I] : **(Array of Float)** Controls maximum values of random parameters.
Allowed values:
$\in \mathbb{R}$: only accept random values smaller than MAX[I]
Array index targets:
1 : torsional terms
2 : CS_6
3 : CS_{12}

MEAN[I] : **(Array of Float)** Controls mean values of the distributions to draw random parameters from.
Allowed values:
$\in \mathbb{R}$: set mean of DIST[I] to MEAN[I]; ignored for uniform and LogUniform distributions
Array index targets:
1 : torsional terms
2 : CS_6
3 : CS_{12}

STDDEV[I] : **(Array of Float)** Controls std. dev. of the distributions to draw random parameters from.
Allowed values:
$\in \mathbb{R}$: set std. dev. of DIST[I] to STDDEV[I]; ignored for uniform and LogUniform distributions
Array index targets:
1 : torsional terms
2 : CS_6
3 : CS_{12}

[seed]

Positions of the variables:

SEED

Details of the variables:

SEED : **(Integer)** Random number generator seed.
Allowed values:
> -1 : set random number generator seed to Python default
> 0 : set random number generator seed to SEED

[genetic_algorithm]

Positions of the variables:

POPSIZE	NGENS	
SELTYPE	NTEL	
CRTYPE	CRRATE	MUTRATE

Details of the variables:

POPSIZE : **(Integer)** Size of the GA population.
Allowed values:
> 0 : set population size to POPSIZE

NGENS : **(Integer)** Number of generations to run the GA for.
Allowed values:
> 0 : set number of generations to NGENS

SELTYPE : **(Integer)** Selection operator of the GA.
Allowed values:
1 : Roulette operator
2 : Rank operator
3 : Tournament operator

NTEL : **(Integer)** Number of elite individuals in the GA population.
Allowed values:
0 : no elitism
> 0 : set number of elite individuals to NTEL

CRTYPE : **(Integer)** Crossover operator of the GA.
Allowed values:
1 : Arithmetic operator
2 : Heuristic operator

CRRATE : **(Integer)** Probability (over 100) of crossover.
Allowed values:
$0 \ldots 100$: set probability of crossover to CRRATE/100

MUTRATE : **(Integer)** Probability (over 100) of mutation.
Allowed values:
$0 \ldots 100$: set probability of mutation to MUTRATE/100

[writetraj]

Positions of the variables:

NSTE NSTP

Details of the variables:

NSTE : **(Integer)** Frequency of writing population energies to the TRE file.
Allowed values:
0 : do not write population energies to the TRE file
> 0 : write population energies to the TRE file every NSTE-th generation

NSTP : **(Integer)** Frequency of writing population parameters to the TRP file.
Allowed values:
0 : do not write population parameters to the TRP file
> 0 : write population parameters to the TRP file every NSTP-th generation

[torsional_scan]

Positions of the variables:

RFRST RSTEP RLST RFCT

Details of the variables:

RFRST : **(Float)** Starting angle of the torsional scan.
Allowed values:
$\in \mathbb{R}$: set starting angle to RFRST degrees

RSTEP : **(Float)** Step angle of the torsional scan.
Allowed values:
$\in \mathbb{R}$: set step angle to RSTEP degrees

RLST : **(Float)** Last angle of the torsional scan (inclusive).
Allowed values:
$\in \mathbb{R}$: set last angle to RLST degrees

RFCT : **(Float)** Force constant of the dihedral restraint of the torsional scan.
Allowed values:
> 0 : set force constant of the harmonic dihedral restraint to RFCT kJ $\mathrm{mol}^{-1}\mathrm{rad}^{-2}$

[minimization]

Positions of the variables:

MALG DX0 DXM NMAX DELE

Details of the variables:

MALG[I] : **(Array of Integers)** Minimization algorithms.
Allowed values:
1 : steepest-descents (more robust, less prone to errors)
2 : conjugate-gradients (less robust, more prone to errors)
Array index targets:
1 : minimization settings for all individuals and all generations
2 : enhanced minimization settings for optimal individual at the end of the run

DX0[I] : **(Array of Floats)** Initial step sizes DX0 in the minimization algorithms.
Allowed values:
> 0 : set DX0 of MALG[I] to DX0[I]
Array index targets:
1 : minimization settings for all individuals and all generations
2 : enhanced minimization settings for optimal individual at the end of the run

DXM[I] : **(Array of Floats)** Maximal step sizes DXM in the minimization algorithms.
Allowed values:
> 0 : set DXM of MALG[I] to DX0[I]
Array index targets:
1 : minimization settings for all individuals and all generations
2 : enhanced minimization settings for optimal individual at the end of the run

NMAX[I] : **(Array of Integers)** Maximum number of steps in the minimization algorithms.
Allowed values:
> 0 : set the maximal number of steps of MALG[I] to NMAX[I]
Array index targets:
1 : minimization settings for all individuals and all generations
2 : enhanced minimization settings for optimal individual at the end of the run

DELE[I] : **(Array of Floats)** Energy-convergence criteria of the minimization algorithms.
Allowed values:
> 0 : sete DELE for MALG[I] to DELE[I]
Array index targets:
1 : minimization settings for all individuals and all generations
2 : enhanced minimization settings for optimal individual at the end of the run

For more details on the algorithmic meaning of these variables, see Chapter 2.

5.2.3 Interaction Function Parameters File (IFP)

The interaction function parameters file (IFP) contains the final values of the optimized parameters, *i.e.* those of the best individual of the GA population, along with the corresponding value of the weighted RMSD. These quantities are listed below identifying comment lines that are compatible with the notation in Equation 3.5.1.

5.2.4 Parameters Trajectory File (TRP)

The parameters trajectory file (TRP) contains the GA-individual parameters for every NSTP-th generation. The data corresponding to each generation is lead by a line containing the generation number, followed by a comment line labelling the parameters in accordance with the notation in Equation 3.5.1 and then one line per individual in which its parameters are listed. Individuals are ranked from best to worst (increasing order on weighted RMSD). This file is intended to allow the user to check on the diversity of the parameters accross the population and their convergence over the generations.

5.2.5 Energy Trajectory File (TRE)

The energy trajectory file (TRE) contains the GA-individual energy profiles for every NSTE-th generation. The data corresponding to each generation is lead by a line containing the generation number, followed by one line per torsional-scan angle where the energies of each individual at the corresponding torsional-scan step are listed. This file is intended to allow the user to check on the diversity of the energy profiles accross the population and their convergence over generations.

5.2.6 Weighted RMSD Trajectory File (TRR)

The weighted RMSD trajectory file (TRR) contains the average and best values of the weighted RMSD accross the population for every generation. This file is useful to inspect the diversity of the population and its convergence over the generations. If the average value converges too fastly, or if its limiting value is too close to the best

value, then more diversity is needed in the population. This can be achieved, for example, by raising the value of the mutation probability, using a less-conservative crossover operator (*e.g.* the heuristic operator) or adjusting the parameters of the probability distribution functions involved in parameter randomization.

Chapter 6

Tutorial

In this chapter, the use of `profilerTools` is illustrated in a more didactic way. To do this, we consider the problem of optimizing the description of torsions of the form CH3-CHm-CHn-CH3 in a united-atoms representation. This involves optimizing the corresponding torsional parameters together with the 1–4 Lennard-Jones parameters that describe the CH3-CH3 interactions. As reference data, quantum-mechanical torsional-scan energy profiles of two alkane molecules containing this dihedraltype are considered: butane (`bu`) and 2-methylbutane (`2mb`); see Figure 6.1. To explore the capabilities of `profilerOpt` and exemplify its usage, we propose several optimization strategies. Two program features are explored in them: the customization of the weights attributed to the reference data (comparing user-specified weights with uniform weights) and the fine control of torsional functional form (comparing a standard periodic dihedral potential with terms of all multiplicities with one with a single term of a chosen multiplicity). Also, the Lennard-Jones part of the optimization problem is formulated in two alternative (and, for these molecules, equivalent) ways. The first considers the direct optimization of the CH3-CH3 nonbonded pairtype. The other considers the optimization of the CS_6, CS_{12} atomic parameters of the CH3 atomtype. This leads to a total of eight scenarios, out of which we consider the following four:

1. Uniform weights; All multiplicities; Pairtype optimization; (Section 6.3.1)
2. User-specified weights; All multiplicities; Pairtype optimization; (Section 6.3.2)
3. Uniform weights; Single multiplicity; Pairtype optimization; (Section 6.3.3)
4. Uniform weights; All multiplicities; Atomtype optimization; (Section 6.3.4)

Before discussing these examples, we describe the organization of the tutorial files in Section 6.1. After that, in Section 6.2, we explain how to obtain torsional-scan trajectories for `bu` and `2mb` using `profilerGen`. Finally, in Section 6.3, we show how to optimize the parameters in the four different scenarios above, using `profilerOpt`.

Throughout this chapter, we assume the reader has access to a Linux terminal emulator and is familiarized with some type of shell. In particular, the terminal snippets in this chapter were written considering the Bash shell. If this is not your case, please adapt the commands to match your terminal configuration.

6.1 File Structure

The tutorial files can be found inside the `examples` directory, which is organized as follows:

```
|--- coords
|   |--- 2mb.g96
|   |--- bu.g96
|--- inp
|   |--- all-terms.inp
|   |--- custom-terms.inp
|--- qm
|   |--- 2mb.dat
```

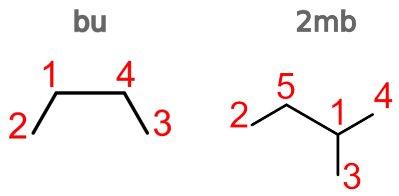

Figure 6.1: Structures and atom indexing of the butane (`bu`) and 2-methylbutane (`2mb`) molecules.

```
|   |--- bu.dat
|--- stp
|   |--- 2mb_atomic.stp
|   |--- 2mb_pairs.stp
|   |--- bu_atomic.stp
|   |--- bu_pairs.stp
|--- wei
    |--- 2mb.dat
    |--- bu.dat
```

The contents of the directories above are:

— `coords`: Initial coordinates of the systems (a single conformation each).

— `inp`: INP files with the input variables for the `profilerOpt` runs.

— `qm`: Quantum-mechanical reference data of the systems.

— `stp`: STP files with topology, reference dihedral and optimization targets of each system.

— `wei`: User-specified weight files.

Note that there are two INP files in the `inp` directory: one for the optimization involving dihedral terms of all multiplicities (`all-terms.inp`) and one for a single multiplicity (`custom-terms.inp`). There are also two STP files for each system in the `stp` directory: one for the optimization of the CH3-CH3 pairtype (`*_pairs.stp`) and one for the CH3 atomtype (`*_atomic.stp`). These files, and also the weight files in the `wei` directory, will be discussed in more details in the following sections.

6.2 Generating Torsional-Scan Trajectories

The first step in this tutorial is to generate torsional-scan trajectories for the two systems. These will serve as input trajectories to `profilerOpt` later on in the parameter optimization step. To perform the torsional scans, only the coordinate and STP files of each system are necessary. The latter must contain the topology part and the `[ refdihedral ]` block specifying the reference dihedral of the scan (see Section 5.2.1). For this tutorial, it is irrelevant which of the two STP files of each system are chosen for performing its scan, since they are equivalent in terms of their force-field parameters. For example, the `stp/bu_pairs.stp` file reads

```
[ defaults ]
; nbfunc comb-rule gen-pairs fudgeLJ fudgeQQ
1        1         atomic    1.0     1.0

[ atoms ]
; type   c6                 c12                cs6                cs12                      q
CH2      7.4684160e-03      3.3965580e-05      4.7238130e-03      4.7419260e-06          0.0000
CH3      9.6138020e-03      2.6646240e-05      6.8525280e-03      6.0308650e-06          0.0000
CH3      9.6138020e-03      2.6646240e-05      6.8525280e-03      6.0308650e-06          0.0000
CH2      7.4684160e-03      3.3965580e-05      4.7238130e-03      4.7419260e-06          0.0000

[ nbpairs ]
;  ai   aj  type
    2    3    2

[ bonds ]
;  ai   aj   type      l0          kb
    1    3    2      0.1530  7.1500e+06
    1    4    2      0.1530  7.1500e+06
    2    4    2      0.1530  7.1500e+06

[ angles ]
;  ai   aj   ak   type   theta0      ka
    3    1    4    2    111.00    530.00
    1    4    2    2    111.00    530.00

[ dihedrals ]
;  ai   aj   ak   al   type      phi          k         mult
    3    1    4    2    1      0.0000      5.9200        3

[ dihedrals ]
; improper dihedrals
;  ai   aj   ak   al   type      phi          k

[ optpairs ]
; idxs
2   3
```

```
[ optdihedrals ]
; idxs
3   1   4   2

[ refdihedral ]
; idx
3   1   4   2
```

and specifies that the torsional scan will be peformed along the CH3-CH2-CH2-CH3 dihedral angle. The optimization blocks `[ opt* ]` are not necessary to run `profilerGen` and will be ignored in this step.

We now proceed to performing the torsional scans. First, it is convenient to create new directories to receive the `profilerGen` output files. Go inside the `examples` directory and run the command:

```
mkdir scans scans/bu scans/2mb
```

to create them for each system. Now, it is time to run `profilerGen`. For butane, this can be done by running the command:*

```
../../profilerGen.py \
    -c coords/bu.g96 \
    -t stp/bu_pairs.stp \
    -dr 0 10 360 \
    -dk 5000 \
    -min 1 \
    -dx0 0.02 \
    -dxm 0.20 \
    -dele 1.0e-09 \
    -nsteps 500000 \
    -op scans/bu/bu
```

To consult the meaning of each command-line option, see Section 4.1. The command above performs a torsional scan of which the steps are $0, 10, 20 \ldots 360$ deg. At each step of the scan, the energy minimization is performed with a steepest-descents algorithm until the (absolute) energy difference between successive steps is less than 1.0×10^{-9} kJ mol^{-1} or until 500000 steps are carried out. The resulting energy profile is reported in `scans/bu/bu.dat` and is also plotted in Figure 6.3. The torsional-scan trajectory is stored in `scans/bu/bu.xyz`. For 2-methylbutane, we proceed in a similar way, replacing `bu` with `2mb`:

```
../../profilerGen.py \
    -c coords/2mb.g96 \
    -t stp/2mb_pairs.stp \
    -dr 0 10 360 \
    -dk 5000 \
    -min 1 \
    -dx0 0.02 \
    -dxm 0.20 \
    -dele 1.0e-09 \
    -nsteps 500000 \
    -op scans/2mb/2mb
```

Once again, the resulting energy profile is reported in `scans/2mb/2mb.dat` and is also plotted in Figure 6.3. The torsional-scan trajectory is stored in `scans/2mb/2mb.xyz`.

6.3 Optimizing the Parameters

In the beginning of the chapter, four scenarios of parameter optimization were listed. Any given scenario differs from the other three only in the choice of the STP and/or INP files and/or in the command-line options given to `profilerOpt`. For this reason, we will discuss in details only the first of these scenarios. For the remaining ones, we will only point out the differences in the corresponding input files or command-line options.

Bear in mind that, as the `profilerOpt` code currently stands, each run may take a couple of hours, even for the rather simple examples in this tutorial. For the sake of speed, you might want to consider parallelizing each run, which can be done with the `-np` command-line option (see **??**). However, this leads to loss of reproducibility, which originates in technical issues regarding random number generation when the `multiprocessing` library is used for parallelization (as is our case). This is a limitation that we intend to address in future versions of the program. For this tutorial, we prioritize reproducibility instead of speed so we report the commands without requesting parallelization. Also, we supply the output files of the runs of each scenario in the `examples_results` directory.

*Backslashes are used to escape the newline character throughout the entire chapter.

6.3.1 Scenario 1

This scenario is defined by uniform weights, torsional terms of all six multiplicities and pairtype optimization. These settings are configured in the STP files `stp/*_pairs.stp` and the in the INP file `inp/all-terms.inp`. In the STP files, pairtype optimization is achieved by specifying the Lennard-Jones optimization targets using the block `[ optpairs ]`. For example, in the STP file of 2-methylbutane (`stp/2mb_pairs.stp`), the fragment that reads

```
[ optpairs ]
; idxs
2   3
2   4
```

indicates that the two CH3-CH3 nonbonded pairs of which the involved atoms are 2, 3 and 2, 4 are the targets of the Lennard-Jones optimization. It is followed by the fragment

```
[ optdihedrals ]
; idxs
4   1   5   2
```

that sets the dihedral angle CH3-CH1-CH2-CH3 formed by the atoms 4, 1, 5 and 2 as the target of the torsional optimization.

The uniform weights and the funcional form of the torsional terms are set in the INP file type. The descriptions of all variables specified in this file are given in Table 5.1, in which cross-references to the meaning of their allowed values are also found. For the sake of thoroughness, we will briefly comment the file that corresponds to this scenario (`inp/all-terms.inp`), proceeding block by block.

First, in

```
[ parameter_optimization ]
; NTORS   NLJ   FTORS
      1     1       1
; TORSNT[I]
; I=  1     2     3     4     5     6
      1     1     1     1     1     1
; LJNT[I]
; I=  1     2
;   CS6  CS12
      1     1
; WTEMP
      0
```

the optimization of both torsional (NTORS) and Lennard-Jones (NLJ) terms is set up. For the former, a standard proper dihedral functional form (FTORS) with terms of all six multiplicities (TORSNT) is used. For the latter, both the attractive and the repulsive components are optimized (LJNT). Finally, if weight files are not supplied when `profilerOpt` is called, the fallback behavior is set to using uniform weights (WTEMP).

In

```
[ seed ]
;   SEED
   19804
```

the random number generator seed is set to 17135. The control of the seed is useful for reproducibility purposes. As long as parallelization is not requested, two `profilerOpt` runs with the same seed will yield the same results.

In

```
[ parameter_randomization ]
;    PINV
       50
; DIST[I]  MIN[I]    MAX[I]     MEAN[I]  STDDEV[I]
; Torsional
        2       0        30        10.0        5.0
; CS6
        2       0         1        8e-3      24e-3
; CS12
        2       0         1        8e-6      24e-6
```

the details of the obtention of random parameters are set up. The distributions (DIST) from which torsional, CS_6 and CS_{12} values are drawn, respectively, are specified along with their mean (MEAN), standard deviation (STDDEV) and minimum (MIN) and maximum allowed (MAX) values. All of these distributions are LogNormal. In the case of torsional parameters, after randomly obtained, they may have their sign reversed with a probability of 0.50 (PINV).

In

```
[ genetic_algorithm ]
; POPSIZE    NGENS
      128     50
; SELTYPE     NTEL
        4        2
;  CRTYPE   CRRATE MUTRATE
        2      100      40
```

the details of the genetic algorithm are set up. The population size (POPSIZE) is set to 128, out of which 2 are elite individuals (NTEL). The tournament selection operator is chosen (SELTYPE), along with the heuristic crossover operator (CRTYPE). The crossover probability (CRRATE) is set to 1, *i.e.* every time two parents reproduce, there is crossover, while the mutation probability (MUTRATE) is set to 0.4. The genetic algorithm is configured to run for 50 generations (NGENS).

In

```
[ writetraj ]
; NSTE   NSTP
     5      5
```

the frequencies of writing individual energy profiles (NSTE) and individual parameters (NSTP) to the TRE and TRP files are specified. In this case, these values are written to disk every 5 generations.

In

```
[ minimization ]
; MALG[I]    DX0[I]  DXM[I]  NMAX[I]     DELE[I]
        1      0.02     0.2       20        1e-2
        1      0.02     0.2   500000        1e-9
```

the minimization algorithm settings are specified. They correspond to two steepest-descents algorithms (MALG) of different precision (DELE) and duration (NMAX). The first one refers to the routine minimization performed for every individual; it runs for at most 20 steps or until the energy difference between successive steps is less than 0.01 kJ mol^{-1}. The second one refers to the final minimization performed only for the optimal individual; it runs for at most 500000 steps or until the energy difference between successive steps is less than 10^{-9} kJ mol^{-1}. The steepest-descents algorithm was chosen instead of the conjugate-gradients one because it is more robust. The conjugate-gradients algorithm may fail when the minimization is performed on individuals with "unbalanced" parameter values, which typically occur on the first generations or after a mutation.

In

```
[ torsional_scan ]
;   RFRST  RSTEP  RLST   RFCT
        0     10   360   5000
```

the torsional-scan details are specified. The scan dihedral angles are set to $0, 10, 20 \ldots 360$ deg and maintained with a dihedral restraint of which the force constant is 5000 kJ mol^{-1} rad^{-2}. It is important that these settings (except for the force constant) match those of the input torsional-scan trajectories. By comparing the contents of this block with the command-line options of the `profilerGen` runs of Section 6.2, one can easily see that this is the case.

With the input files explained, we proceed to the parameter optimization. First, it is once again convenient to create a directory for the output files of `profilerOpt`:

```
mkdir scenario_1
```

To perform the optimization, run the following command:

```
../../profilerOpt.py \
    -r qm/bu.dat qm/2mb.dat \
    -c scans/bu/bu.xyz scans/2mb/2mb.xyz \
    -t stp/bu_pairs.stp stp/2mb_pairs.stp \
    -i inp/all-terms.inp \
    -op scenario_1/opt
```

A complete description of the command-line options can be found in Section 4.2. In the above, note that (*i*) the input trajectories of `profilerOpt` are the output trajectories of `profilerGen` (Section 6.2); (*ii*) the input STP files are those for pairtype optimization and (*iii*) the input INP file is the one where a periodic dihedral with all six multiplicities is specified. During execution, `profilerOpt` will notify in `stdout` the start and completion of each generation. It will also notify each time a new best individual is found, informing the value of its weighted RMSD.

As indicated in Section 4.2, a successful run of `profilerOpt` will create several output files of which the formats are described in Section 5.2. For the sake of thoroughness, we will comment on each of these output files and the interpretation of their contents. The TRR output file (`scenario_1/opt.trr`) reads

```
# GEN              AVG              BEST
    0     2.9678611e+02     1.4876041e+01
    1     9.2343584e+01     1.2555096e+01
...
   49     7.1041770e+01     2.3566693e+00
   50     2.4350660e+01     2.3566693e+00
```

where the ellipsis ... indicates the omission of lines. As indicated by the comments, the first column refers to the generation (0 stands for the initial population), while the second and last columns refer to the average and best (lowest) value of the weighted RMSD, respectively, at the given generation. This file is useful to inspect the diversity of individuals in the GA and also their convergence along the generations.

The TRP output file (`scenario_1/opt.trp`) reads

```
1
#   PHI1            K1   M1 ... PHI6            K6   M6              CS6              CS12             WRMSD
    0.00   -13.6573644    1 ... 0.00     7.8184962    6     1.1244283e-03     4.2265186e-06     1.2555096e+01
    0.00    -9.6352254    1 ... 0.00    -3.1527857    6     4.2770074e-03     2.4808228e-06     1.3642115e+01
...                         ...
    0.00    13.4845343    1 ... 0.00     6.4649437    6     7.1490046e-03     6.3541647e-09     1.7612964e+03
    0.00    12.2490638    1 ... 0.00     6.3990019    6     2.1167276e-02     2.1299433e-08     3.9020127e+03
6
#   PHI1            K1   M1 ... PHI6            K6   M6              CS6              CS12             WRMSD
```

where the ellipsis ... indicates the omission of lines and columns. The fragment reproduced above reports the GA-individual parameters (one individual per line) and the corresponding weighted RMSD of the first generation, as indicated by the top number. The meaning of each column is explained by the comments below the generation number. As indicated by the last lines, the data for the first generation is followed by similarly structured data for the sixth generation (since the frequency of writing to this file was set to five generations in the input variable NSTP).

The TRE output files (`scenario_1/opt_1.tre` and `scenario_1/opt_2.tre`) report the energy profiles of all individuals for all generations. There is one such file for each system, and they are numbered in the same order in which the system files are supplied to `profilerOpt` (in this case, 1 refers to butane and 2 refers to 2-methylbutane). For example, the butane TRE file reads

```
1
#    ANG              IND1 ...        IND128
    0.00     2.2426570e+01 ... 1.5073598e-01
   10.00     1.7133472e+01 ... 1.4272572e+01
...                       ...
  350.00     1.7133447e+01 ... 1.4374804e+01
  360.00     2.2428204e+01 ... 0.0000000e+00
6
#    ANG              IND1 ...        IND128
```

where the ellipsis ... indicates the omission of lines and columns. The fragment reproduced above corresponds to the first generation of the GA, as indicated by the top number. The first column lists the angles of the torsional scan. From the second column to the last, the corresponding energy profiles of each individual are given, one per column. As indicated by the last lines, the data for the first generation is followed by similarly structured data for the sixth generation (since the frequency of writing to this file was set to five generations in the input variable NSTE).

The IFP output file (`scenario_1/opt.ifp`) reads

```
#              CS6              CS12
     2.2379720e-03     3.4437180e-06
#              PHI                K    M
              0.00        0.6443053    1
              0.00        0.8438238    2
              0.00        6.7092465    3
              0.00       -0.7038652    4
              0.00        0.4910962    5
              0.00        0.5522696    6
#            WRMSD
     2.3566693e+00
```

and reports the results of the parameter optimization: the parameters of the optimal individual and the corresponding weighted RMSD. The data is clearly explained by the accompanying comments.

The torsional-scan trajectories of the optimal individual and the corresponding energy profiles are written to the output files with XYZ and DAT extensions, respectively. These files are:

```
scenario_1/opt_1.dat
scenario_1/opt_1.xyz
scenario_1/opt_2.dat
```

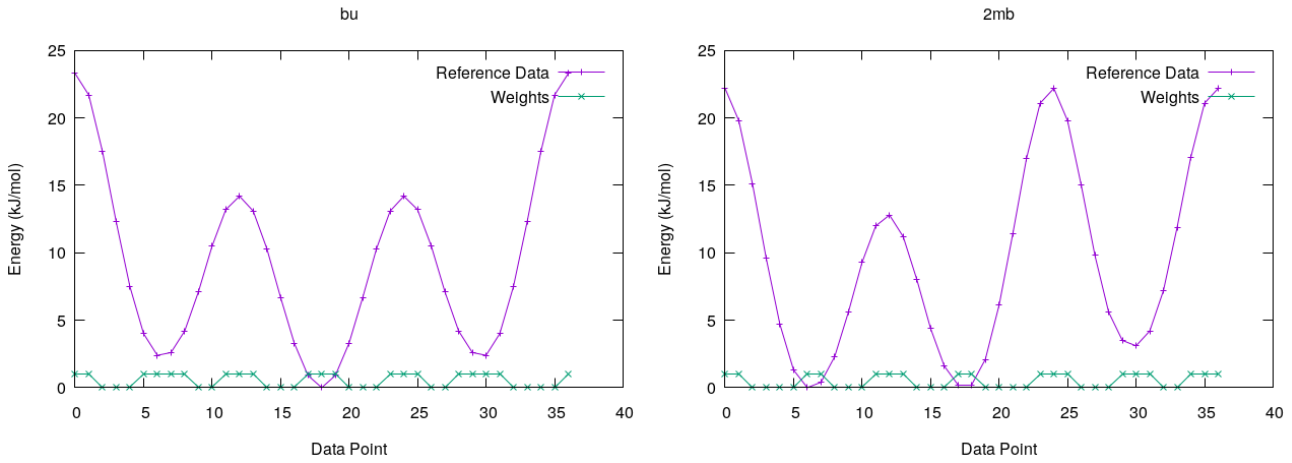

Figure 6.2: Reference data of the optimization and custom weights for each system. The y-axis labels refer only to the reference data; the weights are dimensionless.

```
scenario_1/opt_2.xyz
scenario_1/opt_minim_1.dat
scenario_1/opt_minim_1.xyz
scenario_1/opt_minim_2.dat
scenario_1/opt_minim_2.xyz
```

Each file is identified with the number of the system and with a string indicating the minimization algorithm used in its obtention. The number of the system is defined by the order in which the system files were given as input to `profilerOpt`, like in the TRE files. The minimization algorithm is identified with the "`minim`" string: its presence indicates that the file corresponds to the enhanced minimization algorithm settings, while its absence indicates the less-precise minimization settings involved in every generation of the GA. The optimal energy profiles of `scenario_1/opt_minim_1.dat` and `scenario_1/opt_minim_2.dat` are plotted in Figure 6.3.

6.3.2 Scenario 2

This scenario differs from Scenario 1 in the weights given to the reference data. Instead of using uniform weights, custom weights are supplied *via* the files `wei/bu.dat` and `wei/2mb.dat`. In these files, data points that are close to peaks and valleys are given weight 1, and the remaining ones are given weight 0, *i.e.*, they are irrelevant in the optimization (see Figure 6.2). To perform the optimization, the most important change necessary is to include the weight files in the call to `profilerOpt`, *i.e.*,

```
../../profilerOpt.py \
    -r qm/bu.dat qm/2mb.dat \
    -c scans/bu/bu.xyz scans/2mb/2mb.xyz \
    -t stp/bu_pairs.stp stp/2mb_pairs.stp \
    -i inp/all-terms.inp \
    -op scenario_2/opt \
    -w wei/bu.dat wei/2mb.dat
```

where the last line contains the weight files. In the above, it is assumed that a directory `scenario_2` has been previously created to store the output files. After the run is finished, the optimal parameters reported in the IFP file are

```
#              CS6             CS12
     2.9068136e-03     2.9907985e-06
#              PHI                K    M
               0.00       -0.2056776    1
               0.00        0.1713483    2
               0.00        6.6991703    3
               0.00       -0.0621579    4
               0.00        1.5542293    5
               0.00        1.3870725    6
#            WRMSD
     1.1921891e+00
```

The corresponding energy profiles of `scenario_2/opt_minim_1.dat` and `scenario_2/opt_minim_2.dat` are plotted in Figure 6.3.

6.3.3 Scenario 3

This scenario differs from Scenario 1 in the functional form of the torsional terms. Instead of all summands in Equation 1.3.1, only the term with multiplicity equal to 3 is optimized, while the others are set to zero. This can be achieved by altering the value of the input variable TORSNT in the block `[ parameter_optimization ]` of the INP file. This is done in the `inp/custom-terms.inp` file, where this block reads:

```
[ parameter_optimization ]
; NTORS   NLJ   FTORS
      1     1       1
; TORSNT[I]
; I=  1     2     3     4     5     6
      0     0     1     0     0     0
; LJNT[I]
; I=  1     2
;   CS6  CS12
      1     1
; WTEMP
      0
```

To perform the optimization, the most important change necessary is to use the appropriate INP file in the call to `profilerOpt`, *i.e.*,

```
../../profilerOpt.py \
    -r qm/bu.dat qm/2mb.dat \
    -c scans/bu/bu.xyz scans/2mb/2mb.xyz \
    -t stp/bu_pairs.stp stp/2mb_pairs.stp \
    -i inp/custom-terms.inp \
    -op scenario_3/opt
```

In the above, it is assumed that a directory `scenario_3` has been previously created to store the output files. After the run is finished, the optimal parameters reported in the IFP file are

```
#              CS6              CS12
     2.9293877e-03     2.9476463e-06
#              PHI                 K    M
              0.00         0.0000000    1
              0.00         0.0000000    2
              0.00         7.2805242    3
              0.00         0.0000000    4
              0.00         0.0000000    5
              0.00         0.0000000    6
#            WRMSD
     6.5948745e-01
```

The corresponding energy profiles of `scenario_3/opt_minim_1.dat` and `scenario_3/opt_minim_2.dat` are plotted in Figure 6.3.

6.3.4 Scenario 4

This scenario differs from Scenario 1 in the way that the target of the Lennard-Jones optimization is specified. Instead of the CH3–CH3 pairtype, the CH3 atomtype is targeted in the optimization. Note that these approaches are equivalent for the two molecules considered in this tutorial, since the only 1–4 nonbonded pairtype that they contain is the CH3–CH3 one. If there were any other 1–4 nonbonded pairs involving CH3, this would not be the case, because the atomtype optimization would also affect them, while the pairtype optimization would not.

Atomtype optimization can be achieved by listing the target atoms in the `[ optatoms ]` block of the STP files, which replaces the `[ optpairs ]` block of the previous scenarios. This is done in the STP files `stp/bu_atomic.stp` and `stp/2mb_atomic.stp`. For example, in the 2-methylbutane STP file,

```
[ optatoms ]
; idxs
2   3   4
```

indicates that the atoms 2, 3 and 4 receive the Lennard-Jones parameters of the optimal GA individual. To perform the optimization, the most important change necessary is to use the appropriate STP files in the call to `profilerOpt`, *i.e.*,

```
../../profilerOpt.py \
    -r qm/bu.dat qm/2mb.dat \
    -c scans/bu/bu.xyz scans/2mb/2mb.xyz \
    -t stp/bu_atomic.stp stp/2mb_atomic.stp \
    -i inp/all-terms.inp \
    -op scenario_4/opt
```

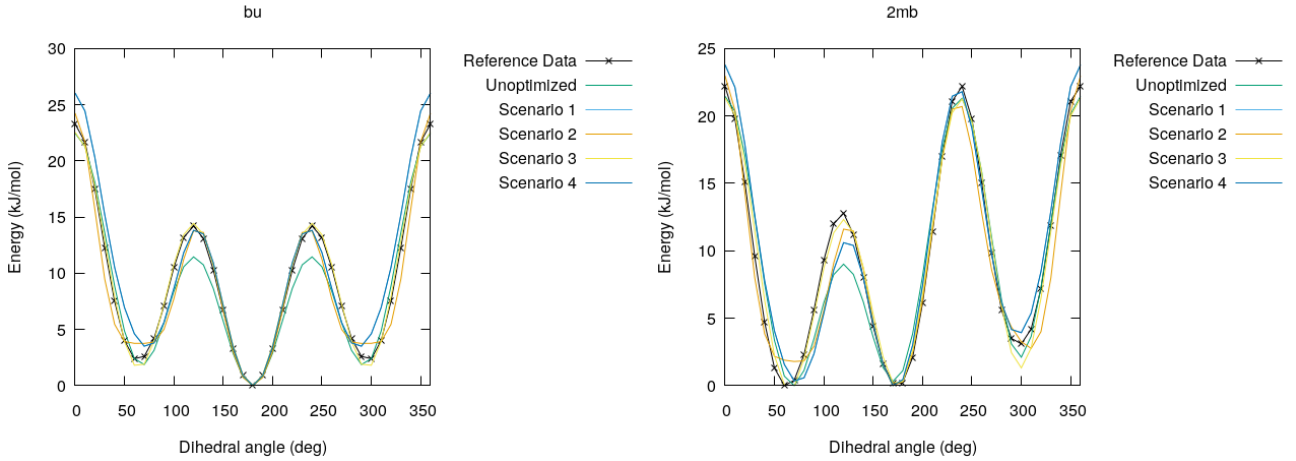

Figure 6.3: Optimal energy profiles of the four scenarios studied in this tutorial, along with the reference and unoptimized data for each system. Note that the lines of Scenario 1 and Scenario 4 superimpose over each other. The unoptimized data is from the energy profiles obtained with `profilerGen` in Section 6.2, using the original parameters of the STP files.

In the above, it is assumed that a directory `scenario_4` has been previously created to store the output files. After the run is finished, the optimal parameters reported in the IFP file are:

```
#               CS6              CS12
     2.2379720e-03     3.4437180e-06
#               PHI                K    M
                0.00        0.6443053    1
                0.00        0.8438238    2
                0.00        6.7092465    3
                0.00       -0.7038652    4
                0.00        0.4910962    5
                0.00        0.5522696    6
#              WRMSD
     2.3566693e+00
```

Note that they are identical to the ones obtained in Scenario 1 (Section 6.3.1), as anticipated above. The corresponding energy profiles of `scenario_4/opt_minim_1.dat` and `scenario_4/opt_minim_2.dat` are plotted in Figure 6.3.