Rust INTDER Manual

Brent R. Westbrook

INTDER is a program for performing coordinate transformations between simple-/symmetry-internal coordinates and Cartesian coordinates. It was originally written in Fortran by Wesley D. Allen and coworkers but has since been translated to Rust by Brent R. Westbrook. This documentation corresponds to the Rust version only.

1 Internal Coordinates

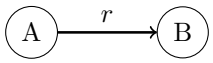
At their most basic, internal coordinates are those defined with reference to other atoms in the same molecule (or system), as opposed to an external set of axes or some other frame of reference. Simple examples of internal coordinates are distances between pairs of atoms and angles between sets of three atoms. A major benefit of using internal coordinates over an external coordinate system like Cartesian coordinates is that internal coordinates require only $3N - 6$ (or $3N - 5$ for linear molecules) coordinates to describe a system fully since the 3 translational and 3 rotational degrees of freedom only have meaning in an external reference frame. N in this case is, of course, the number of atoms in the system. Given the exponential scaling of the number of points required to describe a quartic force field (QFF), or really any potential energy surface (PES), with the number atoms, this difference of 6 coordinates can have a large effect. As a result, using internal coordinates can lead to a substantial decrease in computational cost for such a PES. The types of simple-internal coordinates supported by INTDER are described in the following subsections.

1.1 Simple-Internal Coordinates

1.1.1 Stretch

The bond stretch, referred to as `STRE` in the input file, is the distance between two atoms, as shown in Fig. 1. When the direction matters, the distance is computed from atom `A` to atom `B`, using the equation given in Eqn. 1, where a_i and b_i represent the ith components of the Cartesian coordinates of atoms `A` and `B`, respectively.

Figure 1: A stretch



$$r = \sqrt{\sum_i (b_i - a_i)^2} \tag{1}$$

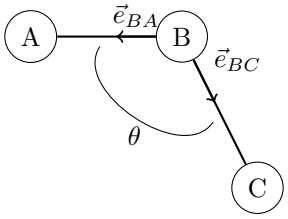
1.1.2 Bend

The bend, referred to as `BEND` in the input file, is the angle between three atoms, as shown in Fig. 2.

The value of the angle, θ is computed using Eqn. 2, where $\vec{e}_{ij}$ represents the unit vector pointing from atom i to atom j.

$$\theta = \text{acos}(\vec{e}_{BA} \cdot \vec{e}_{BC}) \tag{2}$$

Figure 2: A bend

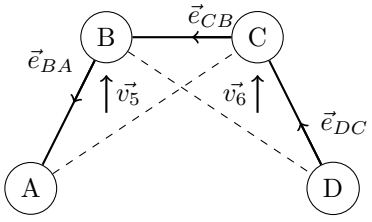


1.1.3 Torsion

The torsional angle, referred to as `TORS` in the input file, is the angle between the planes formed by the overlapping sets of atoms `A`, `B`, `C`, and `B`, `C`, `D`, as shown in Fig. 3. Vectors $\vec{v}_5$ and $\vec{v}_6$ are the vectors normal to the planes ABC and BCD, respectively, found by $\vec{e}_{BA} \times \vec{e}_{CB}$ and $\vec{e}_{DC} \times \vec{e}_{CB}$. To find the angle θ between these two normals, and thus between the planes, perform the following transformations:

$$w_2 = \vec{e}_{BA} \cdot \vec{e}_{CB} \tag{3}$$
$$w_3 = \vec{e}_{DC} \cdot \vec{e}_{CB} \tag{4}$$
$$s_2 = \sqrt{1 - w_2^2} \tag{5}$$
$$s_3 = \sqrt{1 - w_3^2} \tag{6}$$
$$w_4 = \vec{e}_{BA} \cdot \vec{v}_6 \tag{7}$$
$$w_5 = -\vec{v}_5 \cdot \vec{v}_6 \tag{8}$$
$$w = \frac{w_2}{s_2 s_3} \tag{9}$$
$$\theta_0 = \operatorname{asin} w \tag{10}$$
$$\theta = \begin{cases} \theta_0 & w_3 \geq 0 \\ \pi - \theta_0 & w_3 < 0 \end{cases} \tag{11}$$

Figure 3: A torsion



1.1.4 Lin1

TODO

1.1.5 Out

TODO