MSTor: A program for calculating partition functions, free energies, enthalpies, entropies, and heat capacities of complex molecules including torsional anharmonicity

Manual

Jingjing Zheng, Steven L. Mielke
Department of Chemistry and Supercomputing Institute, University of Minnesota, Minneapolis, MN 55455

Kenneth L. Clarkson
IBM Almaden Research Center, San Jose, CA 95120

Rubén Meana-Pañeda and Donald G. Truhlar
Department of Chemistry and Supercomputing Institute, University of Minnesota, Minneapolis, MN 55455

Program version: 2013
Program version date: February 1, 2013
Manual updated date: March 26, 2014

Abstract: *MSTor* is a computer program for calculating gas-phase molecular partition functions and thermodynamic functions (standard state energy, enthalpy, entropy, free energy, and heat capacity at constant pressure) as functions of temperature by the multi-structural approximation with torsional anharmonicity (MS-T). The MS-T approximation accounts for the coupling of torsions to one another and to overall rotation and, to some extent, the coupling between torsions and other vibrational modes. The program also calculates partition functions and thermodynamic functions by the multi-structural local harmonic (MS-LH) approximation or multi-structural local quasiharmonic (MS-LQ) approximation. This program package also includes seven utility codes that can be used as stand-alone programs. One utility program calculates reduced moments of inertia by the method of Kilpatrick and Pitzer, one generates conformational structures, the third and fourth calculate volumes of torsional subdomains defined by Voronoi tessellation either analytically or by Monte Carlo sampling, the fifth and the sixth generates template input files, and the seventh calculates one-dimensional torsional partition functions using the torsional eigenvalue summation method.

1. Introduction

The *MSTor* program is a computer program for calculating partition functions and thermodynamic functions of molecules by the multi-structural approximation with torsional anharmonicity (MS-T). The MS-T method is designed to yield the harmonic-oscillator results in the low-temperature limit and to yield free-rotor results for torsions in the high-temperature limit; furthermore, the harmonic results are adjusted by torsional correction factors that are determined using either uncoupled torsional potential or coupled torsional potential in order to obtain a more accurate approximation in the intermediate-temperature regime. The MS-T method does not require assigning torsions to specific normal modes. Therefore the *MSTor* program can handle cases in which the system has significant coupling, even at the normal mode level, between torsions and has coupling between torsions and other kinds of vibrational motion.

This program package also includes utilities that can help the user to generate the input files for the code. This processes is explain in more detail in the Section 7.

2. Licensing

MSTor is copyrighted. Usage of *MSTor* requires a license, which may be obtained at

http://comp.chem.umn.edu/mstor.

or from the CPC Program Library. Usage of the code implies acceptance of the license provisions, which are:

1. No user or site will redistribute the source code or executable code to a third party in original or modified form without written permission of the principal investigator (Donald G. Truhlar) except as explicitly permitted by the following paragraph (numbered item 2). The licensee has no ownership rights in the software or in any copyrights for the software or documentation through this license.
2. A user/group license entitles the licensee (a person) and his/her research group and collaborators to use the program and share the source or executable code for use within a single research group. A site license enables the licensee (company, organization, or computing center) to install the program and allow access to the executable code to any number of users at that company, organization, or computing center, respectively.
3. Publications resulting from using the software will reference the program. The recommended reference is given in this manual.
4. No guarantee is made that the program is bug-free or suitable for specific applications, and no liability is accepted for any limitations in the mathematical methods and algorithms used within the program.
5. No consulting or maintenance services are guaranteed or implied.

3. Citations for the MS-T methods and the *MSTor* program

MS-T(U) method (based on an uncoupled torsional potential)

1. "Practical Methods for Including Torsional Anharmonicity in Thermochemical Calculations on Complex Molecules: The Internal-Coordinate Multi-Structural Approximation," J. Zheng, T. Yu, E. Papajak, I, M. Alecu, S. L. Mielke, and D. G. Truhlar, Physical Chemistry Chemical Physics **13**, 10885–10907 (2011).

MS-T(C) method (based on a coupled torsional potential)

2. "Quantum Thermochemistry: Multi-Structural Method with Torsional Anharmonicity Based on a Coupled Torsional Potential," J. Zheng and D. G. Truhlar, Journal of Chemical Theory and Computation **9**, 1356-1367 (2013).

***MSTor* program**

3. "*MSTor*: A program for calculating partition functions, free energies, enthalpies, entropies, and heat capacities of complex molecules including torsional anharmonicity," J. Zheng, S. L. Mielke, K. L. Clarkson, and D. G. Truhlar, Computer Physics Communications **183**, 1803-1812 (2012).
4. "*MSTor* version 2013: A new version of the computer code for the multistructural torsional anharmonicity with a coupled torsional potential", J. Zheng, R. Meana-Pañeda, and D. G. Truhlar, Computer Physics Communications **184**, 2032-2033 (2013).

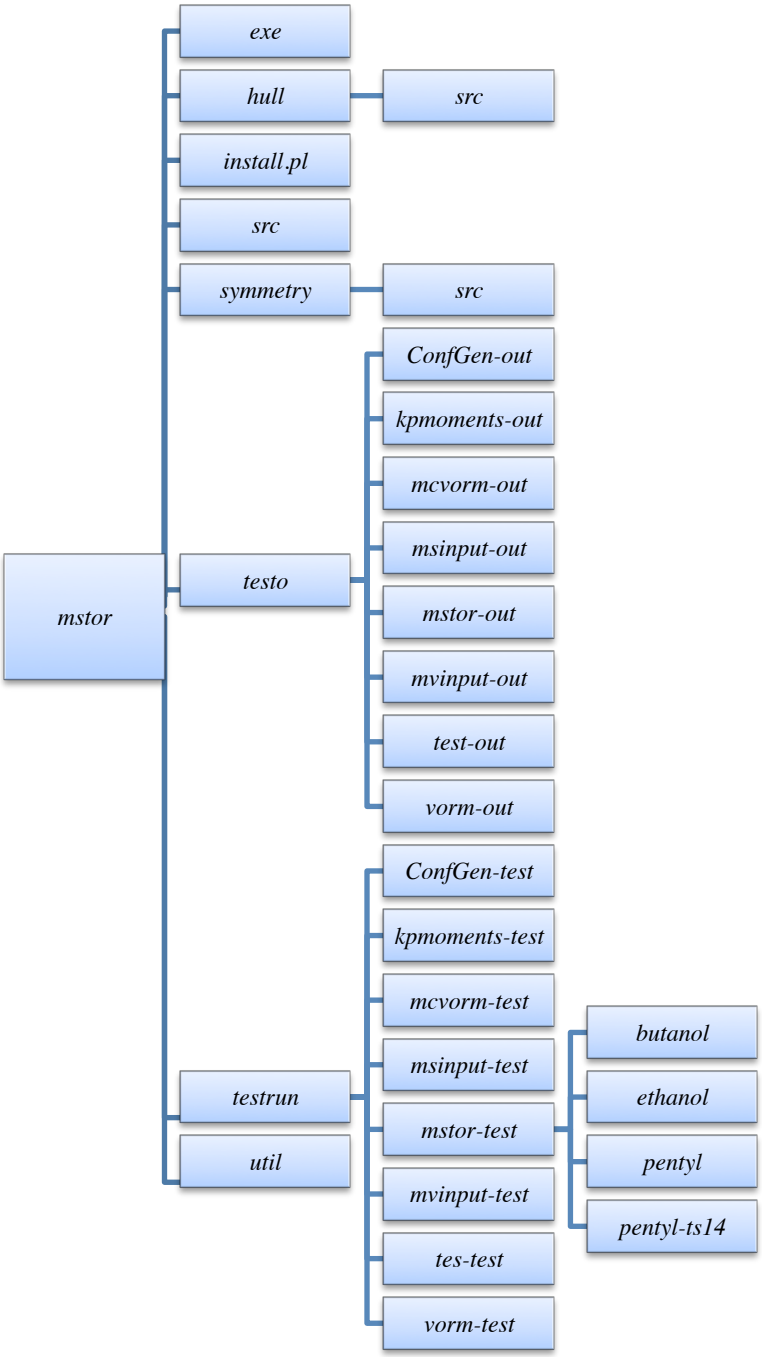
3.1. References for incorporated code

1. The code for evaluating of the point group of symmetry of a structure is from Serguei Patchkovskii (http://www.cobalt.chem.ucalgary.ca/ps/symmetry) and is redistributed under GNU general public license in the *MSTor* package.

4. Distribution and installation

4. 1. Distribution

The distributed *MSTor* package, *mstor.tar.gz*, contains the following directories and files:



exe: This is an empty directory. After successful installation, this directory should contain eight executable files as described in Section 4. C.

hull: This has a subdirectory *src/* that contains the source code of the *hull* program.

install.pl: This is a Perl script that can be used for installation of the *MSTor* program.

src: This directory contains source code for the main program *mstor.exe*

symmetry: This has a subdirectory *src/* that contains the source code of the *symmetry* program.

testo: This directory contains eight subdirectories, *ConfGen-out*, *kpmoments-out*, *mcvorm-out*, *msinput-out*, *mvinput-out*, *mstor-out*, *tes-out*, and *vorm-out*. The eight subdirectories contain output files of test runs for the *mstor.exe* program and the seven utility programs, respectively.

testrun: This directory contains eight subdirectories, *ConfGen-test*, *kpmoments-test*, *mcvorm-test*, *msinput-test, mvinput-test*, *mstor-test*, *tes-test*, and *vorm-test*. The eight subdirectories contain input files of test runs for the *mstor.exe* program and the seven utility programs, respectively. The *mstor-test* directory also has four subdirectories that contain test files for different molecules including one test at a saddle point.

utili: This directory contains the source code of six utility programs.

4. 2. Installation

The installation procedure consists of the following steps:

1) Copy the *mstor.tar.gz* package to the installation directory and then unzip and untar the package.
   tar -xzvf <Return>
2) Go to the *mstor/* directory and run the installation script
   ./install.pl <Return>

The *install.pl* script will search for available Fortran and C compilers and compile the *MSTor* code in the *mstor*/src/ directory. It will also compile the *hull* code in the *mstor*/*hull*/ directory, and it will compile all utility codes in the *mstor*/*utili*/ directory.

3) If the install script runs successfully, the *mstor*/*exe*/ directory will contain ten executables: *ConfGen.exe, kpmoments.exe*, *mcvorm.exe*, *mstor.exe*, *hull.exe*, *msinput.exe*, *mvinput.exe*, *symmetry.exe*, *tes.exe*, and *vorm.exe*.

4) Add the *mstor*/*exe*/ directory to your Linux/Unix PATH variable. This step is mandatory to run *vorm.exe* because *vorm.exe* calls *hull.exe* by assuming the system can find *hull.exe* in the PATH variable. Alternatively, one can copy all the executables to a directory that is already in the Linux/Unix PATH variable.

4. 3. Description of Executables

Executable	Description	Usage
mstor.exe	Main executable for calculating the MS-T partition function, the MS-LH partition function, and thermodynamic functions	*mstor.exe* < input > output
vorm.exe	Utility code to calculate local periodicities (*M* values) using Voronoi tessellation by calling *hull.exe*	*vorm.exe* < input > output
mcvorm.exe	Utility code to calculate local periodicities (*M* values) using the Monte Carlo method	*mcvorm.exe < input > output*
msinput.exe	Utility code to generate a template input file for *mstor.exe*. If the *mvorm.out* file exists, the code also reads the local periodicities from the *mvorm.out* file.	*msinput.exe* < input
mvinput.exe	Utility code to generate a template input file for *vorm.exe* and/or *mcvorm.exe*	*mvinput.exe* < input
hull.exe	Computes the convex hull of a point set in general dimension. It is used by *vorm.exe* to calculate each Voronoi cell volume	see http://www.netlib.org/voronoi/hull.html for more details about using this as a standalone program.
symmetry.exe	Determines the symmetry point group of the molecular structure.	see http://www.cobalt.chem.ucalgary.ca/ps/symmetry for more details, and to use it as a standalone program.
ConfGen.exe	Utility code to generate a set of conformational structures by rotating user specified bonds in an input structure	*ConfGen.exe* < input
kpmoments.exe	Utility code to compute reduced moments of inertia by the method of Kilpatrick and Pitzer	*kpmoments.exe < input > output*
tes.exe	Utility code to calculate 1-D torsional partition functions using the torsional eigenvalue summation method	*tes.exe < input > output*

5. Theoretical background

The total partition function is calculated here by

$$Q_{\text{total}} = Q_{\text{trans}} Q_{\text{elec}} Q_{\text{con-rovib}} \tag{1}$$

where Q_{trans} is the translational partition function, Q_{elec} is the electronic partition function, and $Q_{\text{con-rovib}}$ is the conformational–rovibrational partition function. Equation (1) assumes that the translational partition function and electronic degrees of freedom are separable from the other degrees of freedom. In all equations, the zero of energy for conformational–rovibrational partition functions is at the local minimum of the potential energy function, not at the lowest-energy vibrational state. Notice that this differs from the usual textbook convention (see section 5.8).

The thermodynamic functions for the Gibbs free energy ($G°$), average energy ($E°$), enthalpy ($H°$), and entropy ($S°$) are calculated analytically as

$$G° = -\ln(Q)/\beta + k_B T \tag{2}$$

$$E° = -\frac{\partial \ln(Q)}{\partial \beta} \tag{3}$$

$$H° = E° + P°V \tag{4}$$

$$S° = k_B \ln Q - \frac{1}{T}\left(\frac{\partial \ln Q}{\partial \beta}\right) \tag{5}$$

where $\beta = 1/k_B T$, k_B is the Boltzmann's constant, T is temperature, P is pressure, and V is volume. The degree symbol (°) denotes standard state. The default for the standard state pressure is 1 bar, which equals 100 kPa or 0.987 atm.

Heat capacity at constant pressure (C_p°) is calculated using a four-point central finite differences formula

$$C_p^\circ(T) = \frac{H°(T-2\delta T) - 8H°(T-\delta T) + 8H°(T+\delta T) - H°(T+2\delta T)}{12\delta T} \tag{6}$$

where δT is a step size for temperature.

5. 1. Notations for MS methods

The program calculates two MS approximations: MS-T (multistructural method for torsional anharmonicity) and MS-LH (multistructural local harmonic approximation). These methods were denoted in the original publication by other names as indicated in the chart below. The new names are recommended for future usage.

New name	Original or other name		
MS-T	MS-AS[a]	MS-AS-T[b]	
MS-LH	MS-HO[a]	MS-AS-HO[b]	MS-LQ[c]

[a] J. Zheng, T. Yu, E. Papajak, I. M. Alecu, S. L. Mielke, D. G. Truhlar, Phys. Chem. Chem. Phys. 13 (2011), 10885.
[b] T. Yu, J. Zheng, D. G. Truhlar, Chem. Sci. 2 (2011), 2199.
[c] When one uses scaled frequencies[1] in the MS-LH method, it is actually locally quasiharmonic (LQ or QH) because the scaling accounts not only for systematic errors in the electronic structure method, but also for anharmonicity of the high-frequency modes. In such a case the method may still be called MS-LH, because the formulas are still based on the harmonic oscillator, or it may be called MS-LQ[2], where LQ denotes local quasiharmonic.

The MS-T method can be either based on a coupled torsional potential or an uncoupled torsional potential. To distinguish the method based on different potentials, we use the a specific name for each case as

Method	Potential	Reference
MS-T(U)	Uncoupled torsional potential	3
MS-T(C)	Coupled torsional potential (default)	4

[1] I. M. Alecu, J. Zheng, Y. Zhao, and D. G. Truhlar, J. Chem. Theory Comp. **6**, 2872-2887 (2010).
[2] E. Papajak, P. Seal, X. Xu, and D. G. Truhlar, J. Chem. Phys. **137**, 064110/1-8 (2012).
[3] J. Zheng, T. Yu, E. Papajak, I, M. Alecu, S. L. Mielke, and D. G. Truhlar, Phys. Chem. Chem. Phys. **13**, 10885–10907 (2011).
[4] J. Zheng and D. G. Truhlar, J. Chem. Theory Comp **9**, 1356-1367 (2013)**.**

5. 2. Translational partition function

The translational partition function for an ideal molecular gas is given by

$$Q_{\text{trans}} = \left(\frac{mk_BT}{2\pi\hbar^2} \right)^{3/2} V^\circ \tag{7}$$

where V° is the volume per particle in the standard state, k_B is the Boltzmann's constant, T is temperature, $\hbar$ is Planck's constant divided by 2π, and m is the molecule's mass. For an ideal gas, $V^\circ = k_BT/P^\circ$, where P° is the pressure in the standard state.

Very often we actually quotes the answer as translation partition function per unit volume

$$\Phi_{\text{trans}} = Q_{\text{trans}} / V^\circ \tag{8}$$

Similarly the total partition function per unit volume is

$$\Phi_{\text{total}} = Q_{\text{total}} / V^\circ = \Phi_{\text{total}} Q_{\text{elec}} Q_{\text{con-rovib}} \tag{9}$$

5. 3. Electronic partition function

The electronic partition function is calculated by

$$Q_{\text{elec}} = \sum_i d_i e^{-\varepsilon_i / k_BT} \tag{10}$$

where d_i and ε_i are the degeneracy and energy of the electronic state i, respectively. Here we set the ground electronic state energy as zero ($\varepsilon_1 = 0$). The partial derivative of Q_{elec} with respect to β is

$$-\frac{\partial \ln Q_{\text{elec}}}{\partial \beta} = \frac{1}{Q_{\text{elec}}} \sum_i d_i \varepsilon_i e^{-\varepsilon_i \beta} \tag{11}$$

5. 4. Conformational–rovibrational partition function

5.4.1 MS-T(C) method

When torsions are coupled, we assume that each coupled torsion η for a structure j has the reference potential form, that is,

$$V_{j,\eta} = U_j + \frac{W_{j,\eta}^{(\text{C})}}{2} \left[1 - \cos M_{j,\eta} (\phi_{j,\eta} - \phi_{j,\eta,\text{eq}}) \right]; \quad \frac{-\pi}{M_{j,\eta}} \le \phi_\eta - \phi_{j,\eta,\text{eq}} \le \frac{\pi}{M_{j,\eta}} \tag{12}$$

where U_j is the energy of structure j (where the global minimum is set to 0), $W_{j,\eta}^{(\text{C})}$ is effective coupled barrier height, $M_{j,\eta}$ is a local periodicity parameter, $\phi_{j,\eta}$ is torsional coordinate, and $\phi_{j,\eta,\text{eq}}$ is the torsional coordinate at equilibrium geometry.

For a molecule or a transition state that has J distinguishable structures and t torsions, the conformational–rovibrational partition function according to the MS-T(C) method is

$$Q_{\text{con-rovib}}^{\text{MS-T(C)}} = \sum_{j=1}^{J} Q_{\text{rot},j} \exp\left(-\beta U_j\right) Q_j^{\text{HO}} \prod_{\eta=1}^{t} f_{j,\eta} \tag{13}$$

$$Q_j^{\text{HO}} = \prod_{i=1}^{F} \frac{\exp(-\beta\hbar\omega_{j,i}/2)}{1-\exp(-\beta\hbar\omega_{j,i})} \tag{14}$$

where $Q_{\text{rot},j}$ is the rotational partition function of structure j, β is $1/k_B T$, k_B is Boltzmann's constant, T is temperature, Q_j^{HO} is the usual normal-mode harmonic oscillator vibrational partition function calculated at structure j, $\tilde{f}_{j,\eta}$ is a factor that takes account of torsional anharmonicity, F is number of degrees of freedom for vibrational modes, and $\omega_{j,i}$ denotes the normal-mode vibrational frequency of mode i of structure j. Note that the zero of energy for Q_j^{HO} is at the local minimum of the potential energy function for structure j, not at the zero point level of structure j.

We use the classical expression for the rotational partition function for structure j

$$Q_{\text{rot},j} = \frac{\sqrt{\pi}}{\sigma_{\text{rot,j}}}\left(\frac{2}{\hbar^2\beta}\right)^{3/2} \sqrt{I_{\text{A},j} I_{\text{B},j} I_{\text{C},j}}\ . \tag{15}$$

where $\sigma_{\text{rot},j}$ is the symmetry number of overall rotation, and $I_{\text{A},j}$, $I_{\text{B},j}$, and $I_{\text{C},j}$ are the principal moments of inertia.

If the $f_{j,\eta}$ are set to unity, the partition function $Q_{\text{con-rovib}}^{\text{MS-T(C)}}$ reduces to the multi-structural local-harmonic (MS-LH) partition function. Note that the resulting method is called "local harmonic" but the result is actually quasiharmonic if one uses frequencies scaled[1] to account for anharmonicity (and possibly also accounting for other factors).

Because the torsional anhamronicity correction is based on the coupled torsional potential, the individual $f_{j,\eta}$ is not meaningful and the product of $f_{j,\eta}$ is the correction for torsional anharmonicity for the t coupled torsions,

$$\prod_{\eta=1}^{t} f_{j,\eta} = \left(2\pi\hbar\beta\right)^{t/2} \frac{\prod_{m=1}^{F} \omega_{j,m}}{\prod_{\bar{\bar{m}}=1}^{F-t} \bar{\bar{\omega}}_{j,\bar{\bar{m}}}} \frac{\sqrt{\det \mathbf{D}_j}}{\prod_{\tau=1}^{t} M_{j,\tau}} \prod_{\eta=1}^{t} \exp(-\beta W_{j,\eta}^{(\text{C})}/2) I_0(\beta W_{j,\eta}^{(\text{C})}/2) \tag{16}$$

where $\bar{\bar{\omega}}_{j,\bar{\bar{m}}}$ are torsion-projected normal mode frequencies, $\mathbf{D}_j$ are the Kilpatrick and Pitzer torsional moment of inertia matrices, $M_{j,\tau}$ are local periodicity parameters for uncoupled torsion τ, and I_0 is a modified Bessel function. Note that it is difficult to determine the coupled periodicity parameter $M_{j,\eta}$ so that we use the $M_{j,\tau}$ here instead of $M_{j,\eta}$.

To calculate the thermodynamic functions from the MS-T(C) partition function, we need the partial derivative of the logarithm of the partition function with respect to β:

$$-\frac{\partial}{\partial\beta}\ln Q_{\text{con-rovib}}^{\text{MS-T(C)}}=\frac{1}{Q_{\text{con-rovib}}^{\text{MS-T(C)}}}\sum_{j=1}^{J}\left(Q_{\text{rot},j}\exp\left(-\beta U_j\right)Q_j^{\text{HO}}\prod_{\eta=1}^{t}f_{j,\eta}\right)\left\{\frac{3}{2\beta}+U_j\right.$$
$$\left.+\sum_{m=1}^{F}\frac{\hbar\omega_{j,m}}{2}\left(\frac{1+e^{-\beta\hbar\omega_{j,m}}}{1-e^{-\beta\hbar\omega_{j,m}}}\right)+\sum_{\eta=1}^{t}\left[\frac{W_{j,\eta}^{(\text{C})}}{2}\left(1-\frac{I_1(\beta W_{j,\eta}^{(\text{C})}/2)}{I_0(\beta W_{j,\eta}^{(\text{C})}/2)}\right)-\frac{1}{2\beta}\right]\right\} \quad (17)$$

5.4.2 MS-T(U) method

In the MS-T(U) method, we assume each uncoupled torsion τ has the potential as

$$V_{j,\tau}=U_j+\frac{W_{j,\tau}^{(\text{U})}}{2}\left[1-\cos M_{j,\tau}(\phi_{j,\tau}-\phi_{j,\tau,\text{eq.}})\right];\quad\frac{-\pi}{M_{j,\tau}}\le\phi_{j,\tau}-\phi_{j,\tau,\text{eq.}}\le\frac{\pi}{M_{j,\tau}} \quad (18)$$

where $W_{j,\tau}^{(\text{U})}$ is effective uncoupled barrier height, $\phi_{j,\tau}$ is torsional coordinate, and $\phi_{j,\tau,\text{eq}}$ is the torsional coordinate at equilibrium geometry

For a molecule that has *J* distinguishable structures and *t* torsions, the conformational-rovibrational partition function according to the MS-T(U) method is

$$Q_{\text{con-rovib}}^{\text{MS-T(U)}}=\sum_{j=1}^{J}Q_{\text{rot},j}\exp\left(-\beta U_j\right)Q_j^{\text{HO}}Z_j\prod_{\tau=1}^{t}f_{j,\tau} \quad (19)$$

where Z_j is a factor designed to ensure that the MS-T(U) partition function reaches the correct high-temperature limit, and $f_{j,\tau}$ is an internal-coordinate torsional anharmonicity function that, in conjunction with Z_j, adjusts the harmonic result of structure *j* for the presence of the torsional motion τ.

If the Z_j and $f_{j,\tau}$ are set to unity, the partition function $Q_{\text{con-rovib}}^{\text{MS-T}}$ reduces to the multi-structural local-harmonic (MS-LH) partition function.

The torsional anharmonicity function using uncoupled internal-coordinate approximation for torsion τ at structure *j* is given

$$f_{j,\tau}=\frac{\bar{\omega}_{j,\tau}\sqrt{2\pi\beta I_{j,\tau}}}{M_{j,\tau}}\exp(-\beta W_{j,\tau}^{(\text{U})}/2)I_0(\beta W_{j,\tau}^{(\text{U})}/2) \quad (20)$$

where $\bar{\omega}_{j,\tau}$ is an internal-coordinate torsional frequency, $I_{j,\tau}$ is the torsional moment of inertia calculated by Pitzer's method without coupling between torsions, $M_{j,\tau}$ is a local periodicity parameter, and $W_{j,\tau}^{(\text{U})}$ is an uncoupled effective barrier height. These parameters are interrelated as

$$W_{j,\tau}^{(\text{U})}=\frac{2I_{j,\tau}\bar{\omega}_{j,\tau}^2}{M_{j,\tau}^2} \quad (21)$$

The factor Z_j is given by

$$Z_j=g_j+(1-g_j)Z_j^{\text{int}}Z_j^{\text{coup}} \quad (22)$$

where Z_j^{int} replaces the normal-mode vibrational partition function in the high-T limit by internal-coordinate ones, and Z_j^{coup} replaces the uncoupled moments of inertia for individual torsions by values that account for their coupling. $g_j \to 1$ at low T where the effects of rotational-vibrational coupling are minimal, and $g_j \to 0$ at high T. The Z_j^{int} are given by

$$Z_j^{\text{int}} = \frac{\prod_{m=1}^{F-t} \overline{\overline{\omega}}_{j,m}^{-1} \prod_{\tau=1}^{t} \overline{\omega}_{j,\tau}^{-1}}{\prod_{m=1}^{F} \omega_{j,m}^{-1}} \tag{23}$$

where F is the number of vibrational degrees of freedom, $\omega_{j,m}$ is the frequency of normal mode m. The factors Z_j^{coup} are equal to

$$Z_j^{\text{coup}} = \left(\frac{\left|\det \mathbf{D}_j\right|}{\prod_{\tau=1}^{t} I_{j,\tau}} \right)^{1/2} \tag{24}$$

The expression used for g_j to enforce the correct limits is

$$g_j = \left(\prod_{\tau=1}^{t} \tanh \frac{\sqrt{2\pi k_{j,\tau}\beta}}{M_{j,\tau}} \right)^{1/t} \tag{25}$$

where $k_{j,\tau}$ is the force constant for torsion τ at structure j in internal coordinates.

The partial derivative of the MS-T partition function with respect to β is

$$-\frac{\partial}{\partial\beta} \ln(Q_{\text{con-rovib}}^{\text{MS-T(U)}}) = \frac{1}{Q_{\text{con-rovib}}^{\text{MS-AS}}} \sum_{j=1}^{J} \left\{ e^{-\beta U_j} Q_{\text{rot},j} Q_j^{\text{HO}} Z_j \prod_{\tau=1}^{t} f_{j,\tau} \left(\frac{3}{2\beta} + U_j \right.\right.$$

$$+ \sum_{m=1}^{F} \frac{\hbar\omega_{j,m}}{2} \frac{1+e^{-\beta\hbar\omega_{j,m}}}{1-e^{-\beta\hbar\omega_{j,m}}}$$

$$- \left(g \frac{(1 - Z_j^{\text{int}} Z_j^{\text{coup}})}{2t\sqrt{\beta} Z_j} \left\{ \sum_{\tau=1}^{t} \frac{\overline{\omega}_{j,\tau}\sqrt{2\pi I_{j,\tau}}}{M_{j,\tau}} \frac{\operatorname{sech}^2\left(\overline{\omega}_{j,\tau}\sqrt{2\pi I_{j,\tau}\beta} / M_{j,\tau}\right)}{\tanh\left(\overline{\omega}_{j,\tau}\sqrt{2\pi I_{j,\tau}\beta} / M_{j,\tau}\right)} \right\} \right)$$

$$\left.\left. + \sum_{\tau=1}^{t} \left(\frac{I_{j,\tau}\overline{\omega}_{j,\tau}^2}{M_{j,\tau}^2} - \frac{1}{2\beta} - \frac{I_{j,\tau}\overline{\omega}_{j,\tau}^2}{M_{j,\tau}^2} \frac{I_1(\beta I_{j,\tau}\overline{\omega}_{j,\tau}^2 / M_{j,\tau}^2)}{I_0(\beta I_{j,\tau}\overline{\omega}_{j,\tau}^2 / M_{j,\tau}^2)} \right) \right) \right\} \tag{26}$$

Note that the zero-point energy corresponding to the MS-T partition function is

$$E_0^{\text{MS-LH}} = \min_j \left\{ E_{j,0}^{\text{HO}} + U_j \right\} \tag{27}$$

where $E_{j,0}^{\text{HO}}$ is the harmonic oscillator zero-point energy of structure j with the zero of energy at its local minimum, and which is given by

$$E_{j,0}^{\mathrm{HO}} = \frac{\hbar}{2}\sum_{m=1}^{F}\omega_{j,m} \tag{28}$$

5. 5. Voronoi scheme for *M* values

When torsions are strongly coupled together, it is sometimes impossible to assign integer $M_{j,\tau}$ values for each torsion. We divide the *t* torsions into two types: nearly separable (NS) and strong coupled (SC). We use the notation $\mathrm{NS:SC} = t_{\mathrm{NS}} : t_{\mathrm{SC}}$ to denote that t_{NS} torsions are treated as nearly separable and t_{SC} torsions are treated as strongly coupled. In general, the strongly coupled coordinates may be further partitioned into two or more subspaces, with each subspace involving only those coordinates that are strongly coupled to each other.

Each of the strongly coupled subspaces is treated by Voronoi tessellation separately. Voronoi tessellation divides a space into cells around a discrete set of points. In the applications considered here, the space to be tessellated is described by the dihedral angles $\phi_1,\phi_2,\cdots\phi_{t_{\mathrm{SC}}}$ and the points correspond to structures. Each cell corresponds to a specific structure and consists of all torsional configurations closer to this structure than to any other structure when only the t_{SC} strongly coupled degrees of freedom are considered. We include two kinds of points. (i) the coordinates (sets of dihedral angles) of the distinguishable structures; and (ii) is the coordinates of minima of the potential energy function that correspond to indistinguishable structures but where the angles are determined by considering the torsional symmetries. We label the points $\tilde{j}$ = 1, 2, ..., $\tilde{J}$, where $\tilde{J}$ is greater than or equal to *J*. For example, if we were treat the torsions of propane as strongly coupled (actually they are nearly separable), we would have $J = 1$ and $\tilde{J} = 9$. We label the points as $\tilde{j} = 1, 2, \ldots, J$ for distinguishable structures and $\tilde{j} = J+1, J+2, \ldots, \tilde{J}$ for indistinguishable structures. One may then calculate the volume $\Omega_{\tilde{j}}^{\mathrm{SC}}$ of each cell and associate that volume with that point.

By definition, the volume of the SC torsional subspace neglecting indistinguishability of identical particles is

$$\Omega^{\mathrm{SC,tot}} = \sum_{\tilde{j}=1}^{\tilde{J}}\Omega_{\tilde{j}}^{\mathrm{SC}} = (2\pi)^{t_{\mathrm{SC}}} \tag{29}$$

After accounting for indistinguishability, the total volume of the SC torsional subspace may be calculated by

$$\Omega^{\mathrm{SC}} = \sum_{\tilde{j}=1}^{J}\Omega_{\tilde{j}}^{\mathrm{SC}} \tag{30}$$

Whenever we use Voronoi tessellation we assume that the torsional subspace is so strongly coupled that we cannot assign $M_{j,\tau_{\mathrm{SC}}}$ by considering each torsion separately;

then we replace all $M_{j,\tau_{\mathrm{SC}}}$ for strongly coupled torsions of a given j by a single M_j^{SC} equal to

$$M_j^{\mathrm{SC}} = \frac{2\pi}{\left(\Omega_j^{\mathrm{SC}}\right)^{1/t_{\mathrm{SC}}}} \tag{31}$$

5. 6. Low-temperature limits

In this section, we use the common thermodynamics convention where temperature is written as a subscript. Then

$$G_T^\circ = -\ln(Q)/\beta + k_B T \tag{32}$$

$$E_T^\circ = -\frac{\partial \ln(Q)}{\partial \beta} \tag{33}$$

$$H_T^\circ = E_T^\circ + PV = E_T^\circ + RT = \varepsilon(T) + RT \tag{34}$$

$$S_T^\circ = k_B \ln Q - \frac{1}{T}\left(\frac{\partial \ln Q}{\partial \beta}\right) \tag{35}$$

Consider very low temperature ($T \to 0$), where only the ground state, with energy ε^{G} and degeneracy d_0, is important. Note that if the structure that has the lowest zero-point-inclusive energy is the same as the structure that has the lowest zero-point-exclusive energy, then ε^{G} is the zero point vibrational energy of that structure. Otherwise ε^{G} is given by eq. (18). Then, where all arrows refer to the limit as temperature goes to zero, we have:

$$Q \to d_0 \exp\left(-\beta\varepsilon^{\mathrm{G}}\right) \to 0 \tag{36}$$

Therefore

$$\ln Q \to -\beta\varepsilon^{\mathrm{G}} + \ln d_0 \tag{37}$$

$$-\frac{\partial}{\partial \beta}\ln Q \to \varepsilon^{\mathrm{G}} \tag{38}$$

Substituting eqs. (29)–(31) into eqs. (25)–(27) yields the following low-T limits:

$$G_T^\circ \to \varepsilon^{\mathrm{G}} \tag{39}$$

$$E_T^\circ \to \varepsilon^{\mathrm{G}} \tag{40}$$

$$H_T^\circ \to \varepsilon^{\mathrm{G}} \tag{41}$$

Note that most statistical mechanics textbooks use a different zero of energy, at the ground state level. We label partition functions with that choice as $\tilde{Q}$ where

$$\tilde{Q} \equiv Q \exp\left(\beta \varepsilon^{\mathrm{G}}\right) \tag{42}$$

and we note that the limits of eqs. (30)–(32) would be different with that different choice of the zero of energy

Equation (28) yields

$$S_T^\circ \to \ln d_0 \tag{43}$$

which is consistent with the third law of thermodynamics.

5. 7. Low-temperature limits

Throughout this manual we use the language of stable structures. However the code can also be applied to transition structures. Each stable structure has $3N - 6$ (N is number of atoms) frequencies whereas calculations for transition states are based on the 3N – 7 real frequencies of each transition structure (i.e., each saddle point).

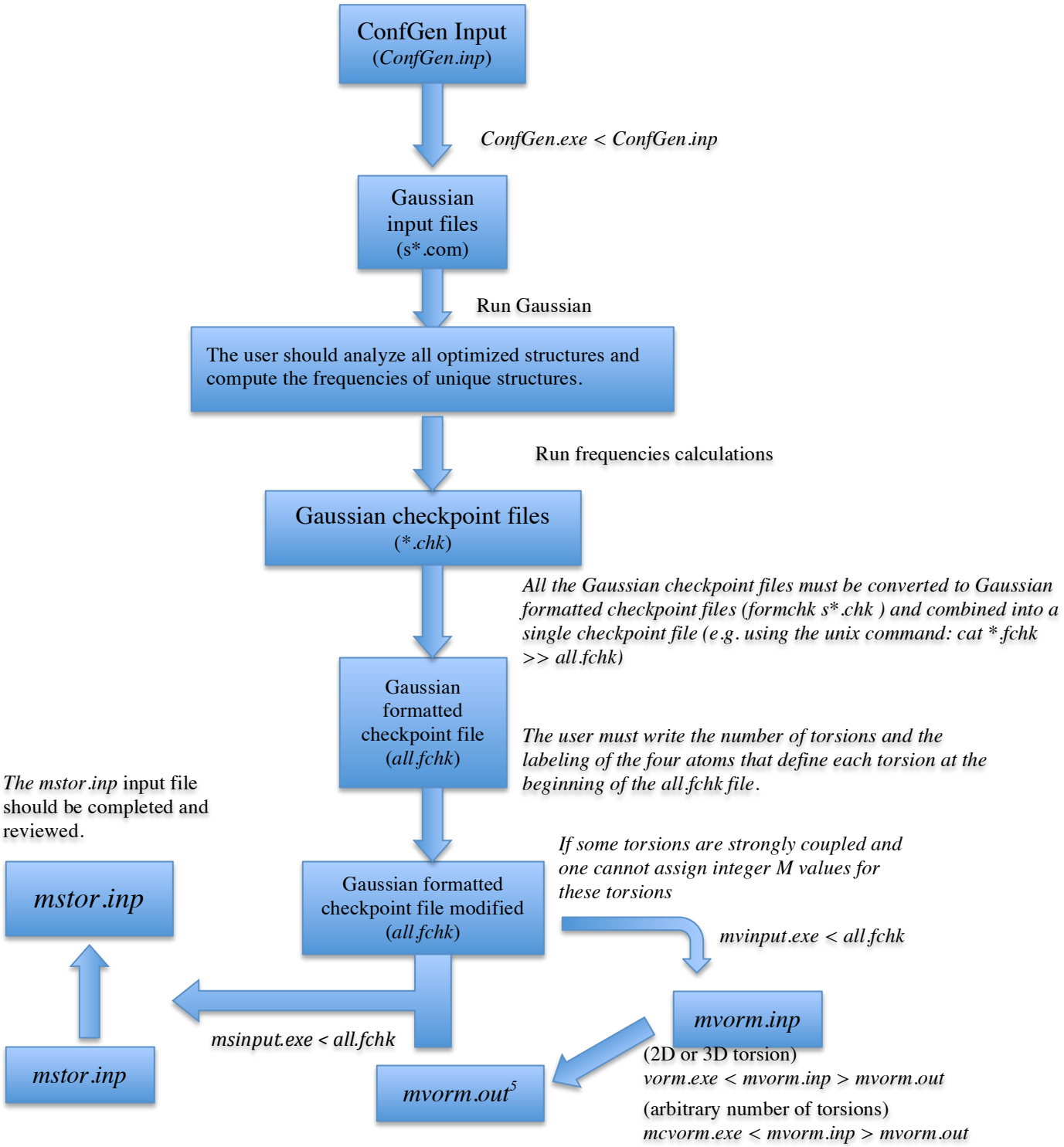
The code determines the system is stable structures or transition structures based on the number of imaginary frequencies.

5. 8. Thermodynamics convention

For thermodynamic calculations based on electronic structure theory, the most convenient zero of energy is the lowest equilibrium (zero-point-exclusive) one. This choice is used throughout the manual and the code. However, almost all textbooks and compilations of the thermodynamic data in the literature set the zero of energy at the zero point level. With this convention the partition function geos to the degeneracy of the ground electronic state at 0 K and enthalpy and free energy go to 0 at 0 K. For the convenience of users, we also print Q, E, H, and G with this thermodynamic convention.

6. Guidelines to generate the *MSTor* input file with *Gaussian* output files

This section describes the steps to generate the *MSTor* input file using some utility codes in combination with the *Gaussian* software package.[5] The following flowchart shows one process that the user could follow:



[5] Frisch, M. J. *et al*. Gaussian 03/09; Gaussian, Inc.: Wallingford, CT, 2003/2009

In first step, the user should carry out a search of all conformations that can be obtained by internal rotation of the various fragments of the initial structure. Using the Cartesian coordinates of the initial structure and the definition of each torsion angle as an input file (denoted in the flowchart as *ConfGen.inp*), the *ConfGen* utility generates the Gaussian input files of the different conformers. The generated structures, obtained by all possible combinations of the values of the dihedral angles specified by the user in the input file, should them be optimized by running the *Gaussian* software package.

In the second stage, all the optimized structures should be analyzed. The main goals are to remove identical structures and to ensure that only one enantiomeric form is included (in the cases where both exist). In the next step, the frequencies of these unique conformations should be calculated and the *Gaussian* checkpoint file (including the Cartesian coordinates, the energy, and the Hessian) should be saved and subsequently formatted using the command *formchk s*.chk*.

Once all *Gaussian* formatted checkpoint files are in the same directory, the user must combine them into a single file (using, for instance, the unix command: cat *.fchk >> all.fchk) and add the number of torsions that are to be treated with the Voronoi tessellation in the first line followed by the four atom numbers that define each torsion (see Subsect 6.6). This file has two uses. On the one hand, it is the input file for the *mvinput* utility, which generates in turn the input file for computing the M_j values using Voronoi tessellation (denoted in the flowchart as *mvorm.inp*) and on the other hand it is used together with the file that contains the M_j values of the different structures (*mvorm.out*), by the utility called *msinput* to generate the *MSTor* input file (denoted as *mstor.inp*).

The *mvinput* utility generates the input file (*mvorm.inp*) for the code that computes the M_j values (*mcvorm.exe* and/or *vorm.exe*) by taking into account the symmetry of each conformer, so that if one structure has an enantiomer, this is also included in the *mvorm.inp* file. The user must always be very careful using this file because there are some cases that deserve special attention (e.g., when it is not possible to interconvert between enantiomers by internal rotation or when several indistinguishable structures located in the full torsion space should be included). These cases are discussed in ref 6.

Indeed, the generated *mstor.inp* file cannot be used directly as an input file of the *MSTor* program because many parameters are not included (e.g., the definition of the non-

---

[6] T. Yu, J. Zheng, and D. G. Truhlar Phys. Chem. Chem. Phys. **14**, 482-494 (2012).

redundant internal coordinates) and some of them are assigned some temporary numbers (e.g., the temperatures).

7. Input files

7. 1. Input files for *mstor.exe* executable

Two input files are required for running the *mstor.exe* executable. One file contains all the information except Hessians, and the other one has the Hessians of all structures. The former will be referred to as the main input file and the latter as the Hessian input file. Note that the name of the Hessian input file is fixed as *hess.dat*.

The main input file consists of several sections. Each section name begins with the symbol $. At the end of each section, a keyword "END" must be given to indicate the end of this section. Each section consists of several keywords. All keywords and section names are case insensitive and any lines beginning with # symbols are comment lines.

Note that each line of input files should be equal or less than 80 characters and the characters after column 80 are ignored by the program.

7. 1. A. Description of sections of the main input file

Section Name	**Description**
$General	General data to describe the system
$Intdef	Defines the non-redundant internal coordinate
$Temp	List of temperature
$Structure #	Gives the geometry, energy, *M* values, and weight for each structure. Here # is an integer to label the structure

7. 1. B. Glossary of keywords in $GENERAL section

The section $GENERAL is for the general information about the system, e.g., the number of atoms, the number of structures, and so on. The following keywords can be used in this section.

ATM

ATM is a keyword to use 1 atmosphere pressure for the standard state pressure. By default, if ATM is not specified, the program uses 1 bar.

BAR

BAR is a keyword to use 1 bar for the standard state pressure. This is the default, and the specification of defaults in the keyword list is optional.

COUPLED

COUPLED is a keyword to use the MS-T(C) method that is based on coupled torsional potential. This is default.

DELTAT

DELTAT is a keyword for specifying the temperature interval (in degrees Kelvin) of eq. (6) for calculating the heat capacity by using the four-point central finite difference formula. The default value is 1 K.

Example:

```
DELTAT      1
```

ELEC

ELEC is a keyword for specifying the degeneracy of the electronic states and their corresponding energies (in hartrees), which are used for calculating the electronic partition function. The ground electronic state has been chosen as the zero of energy. If low-lying excited states exist, they need to be included. We assume that all conformational structures have the same electronic partition function.

Example:

```
ELEC
  1   0.0
END
```

This example indicates that the ground electronic state is a singlet state and that this molecule has no low-lying excited states.

The example below shows a doublet ground electronic state with a low-lying doublet excited state (higher in energy than the ground state by 0.00064 hartrees).

Example:

```
ELEC
  2  0.0
  2  0.00064
END
```

The default, if ELEC is not specified, is "1 0.0".

FREQSCALE

FREQSCALE is a keyword for specifying a frequency-scaling factor. This factor scales all the frequencies used in the calculation. The default value is 1.0. When frequencies are scaled, the harmonic approximation becomes quasiharmonic, but the results are still labeled LH because the H in LH denotes the use of harmonic formulas, not necessarily harmonic frequencies. Note that there are two ways to carry out a quasiharmonic calculation (i) read the scaled Hessians (scaled by the squares of FREQSCALE) and set FREQSCALE equal to one, or (ii) read the harmonic Hessians and scale them with FREQSCALE. This manual assumes that the user is using option (ii).

Example:

```
FREQSCALE  0.970
```

NATOMS

NATOMS is a keyword for specifying the number of atoms in the system. There is no default; NATOMS must be specified.

Example:

```
NATOMS 15
```

NSTR

NSTR is a keyword for specifying the number of structures included in the calculation. This number must be equal to the number of $Structure sections in the input file.

Example:

```
NSTR  15
```

There is no default; NSTR must be specified.

NTOR

NTOR is a keyword for specifying the number of torsions treated with torsional corrections. Note that NTOR can be smaller than the total number of torsions in the studied molecule if one wants to treat some of them using a harmonic approximation.

Example:

```
NTOR  4
```

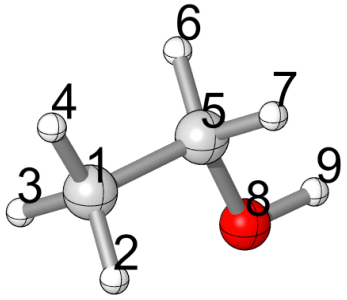
There is no default; NTOR must be specified.

UNCOUPLED

UNCOUPLED is a keyword to use the MS-T(U) method that is based on uncoupled torsional potential.

7. 1. C. Description of $INTDEF section

The section $INTDEF defines the non-redundant internal coordinates. The allowed types are stretches (*i*-*j*), bends (*i*-*j*-*k*), linear bends (*i*=*j*=*k*), and dihedral angles (*i*-*j*-*k*-*l*) where *i*, *j*, *k*, and *l* are atom labels. Each torsion should only be represented by one dihedral angle, and the dihedral angles must be at the end of the list. The example below shows how to define the non-redundant internal coordinates for the ethanol molecule shown below.



Example:

```
$INTDEF
   2-1 3-1 4-1 5-1 6-5 7-5 8-5 9-8
   2-1-4 2-1-3 2-1-5 3-1-4 4-1-5 1-5-6 1-5-7
   6-5-7 6-5-8 7-5-8 5-8-9
   4-1-5-8 9-8-5-1
END
```

Note that the dihedral angles for torsions should be always put in the end.

7. 1. D. Description of $TEMP section

The section $TEMP specifies a list of temperatures. Each line can list multiple temperatures (using integer or floating point format) and this section can have multiple lines. Note that the maximum number of temperature allowed in this section is 100.

Example:

```
$TEMP
   200 298.15 300 400 600 1000
   1500 2000 2400 3000
END
```

7. 1. E. Description of $STRUCTURE section

The section $STRUCTURE specifies the information about each structure involved in the MS-T treatment. The number of $STRUCTURE sections must be the same as that specified by the NSTR keyword in the $GENERAL section. For each $STRUCTURE section, a number must follow $STRUCTURE to distinguish it from the other $STRUCTURE sections, i.e., $STRUCTURE 1. The following keywords can be used in these sections.

ENERGY

ENERGY is a keyword to specify the relative energy of the structure in kcal/mol. The global minimum is usually set to the zero of energy. Note that this energy is zero-point exclusive. The default value is zero.

Example:

```
ENERGY 0.077
```

GEOM

GEOM is a keyword to specify the Cartesian coordinates of the structure. The unit of the Cartesian coordinates is Ångstroms.

Example:

```
GEOM
6       -1.206324   -0.240211   -0.020995
1       -1.264318   -0.787854   -0.963519
1       -1.258873   -0.958451    0.795948
1       -2.072624    0.416915    0.056016
6        0.082734    0.557582    0.046385
1        0.126121    1.277301   -0.769964
1        0.133157    1.122438    0.982766
8        1.230304   -0.257043   -0.105397
```

```
1        1.235647   -0.918227    0.589594
END
```

Note that the first number in each line is the atomic number. For each structure, the order of the atoms should be the same.

One can specify an isotopic mass (in amu) after the coordinates for an atom. If there is nothing after the coordinates, the program uses the default mass, which is the mass of the most abundant isotope. One only needs to specify non-default isotopic masses in the first structure, and anything after the coordinates in the other structures will be ignored. Below is an example for specifying the deuterium mass in the CH_3CH_2OD molecule.

Note that an amu is a universal atomic mass unit equal to 1/12 the mass of a carbon-12 atom. The program uses the following conversion factor: 1 amu = 1822.888 atomic units of mass.

Example:

```
GEOM
6        -1.206324   -0.240211   -0.020995
1        -1.264318   -0.787854   -0.963519
1        -1.258873   -0.958451    0.795948
1        -2.072624    0.416915    0.056016
6         0.082734    0.557582    0.046385
1         0.126121    1.277301   -0.769964
1         0.133157    1.122438    0.982766
8         1.230304   -0.257043   -0.105397
1         1.235647   -0.918227    0.589594        2.014101777
END
```

MTOR

MTOR is a keyword to specify the $M_{j,\tau}$ values. The order of values should be consistent with the order of the dihedral angles defined in the $INTDEF section.

Example:

```
MTOR
   3     3
END
```

If several torsional modes are strongly coupled, one could use Voronoi tessellation to determine the $M_{j,\tau}$ values by using the utility code *vorm.exe* or *mcvorm.exe*. One runs the utility code separately and then inputs the noninteger $M_{j,\tau}$ values here.

Example:

```
MTOR
  2.532   2.532      3       3
END
```

Note that there is no default value for this keyword.

ROTSIGMA

ROTSIGMA is a keyword to specify the symmetry number of overall rotation for the structure given in the $STRUCTURE section. The default value is 1.

Example:

```
ROTSIGMA  2
```

WEIGHT

WEIGHT is a keyword to specify the number of structures that have the same energy and vibrational frequencies as that specified in GEOM. For example, if the geometry given in GEOM has a mirror image structure, one can specify a weight of 2 and only include one of the structures. The default value is 1.

Example:

```
WEIGHT 2
```

7. 1. F. Description of the *hess.dat* input file

The *hess.dat* file contains Hessians for the structures given in the $STRUCTURE sections in the main input file. The only section name is $HESS in *hess.dat* file. Each $HESS should be numbered consistently with the $STRUCTURE sections in the main input file. The Hessian should be given as in upper triangular packed form in unscaled Cartesian coordinates in hartree/bohr^2. Note carefully the ordering of the matrix elements. Each line can have an arbitrary number of elements except that each line cannot exceed 80 characters.

Example:

$HESS 1
F11 F12 F22 F13 F23 F33
F14 F24 F34 F44 F15 F25
F35 F45 F55 F16 F26 F36
F46 F56 F66
END

7. 2. Input files for the *ConfGen.exe* executable

The executable *ConfGen.exe* generates a list of conformational structures by rotating a set of user-specified bonds of an input structure over a specified grid. The program writes each generated structure to a file in *Gaussian* input format.

Example:

```
19      2       ! number of atoms and rotating bonds
C       -10.440384  -1.363155  -2.523127  ! Cartesian coordinates
……
# TORSION 1 DEFINITION (COMMENT LINE)
5 8     !define rotation axis as the bond between atom 5 and atom 8
7       !number of atoms in one fragment
1 2 3 14 5 6 7  ! atom numbers of this fragment
3               ! number of values that the dihedral angles take
0.0 120.0 -120.0        ! angle values by which the torsion will be rotated
#TORSION 2 DEFINITION (COMMENT LINE)
8 11    !define rotation axis as the bond between atom 8 and atom 11
10      !number of atoms in one fragment
 1 2 3 14 5 6 7 8 9 10  ! atom numbers of this fragment
3                       ! number of values that the dihedral angles take
0.0 120.0 -120.0        ! angle values by which the torsion will be rotated
%nproc=8                ! number of processors for Gaussian job
%mem=600mb              ! memory
# Method/Basis          ! route section
 0 1                    ! charge and multiplicity
Extbasis                ! line at the end of the Gaussian input file for
other options (such as using external basis sets)
```

It should be noted that this utility code removes all the generated structures that show an interatomic distance smaller than 0.5 Å.

7. 3. Input files for the *kpmoments.exe* executable

The executable *kpmoments.exe* calculates reduced moments of inertia by the method of Kilpatrick and Pitzer (KP). The input file for *kpmoments.exe* is illustrated by the following example.

Example: (1-butanol)

```
15 5        ! number of atoms and number of frames
```

```
      0.00000000    0.00000000    0.00000000  1.00782503207      1
      0.00000000    0.00000000    0.95000000  15.99491461956     1
      1.32452586    0.00000000    1.40960337  12.0               2
      2.11134566    1.20846054    0.96134036  12.0               3
      3.52426612    1.22340239    1.50755872  12.0               4
      4.31853757    2.43052798    1.05637017  12.0               5
      5.32525047    2.41970290    1.46196246  1.00782503207d0    5
      1.26090910   -0.01952514    2.49366282  1.00782503207d0    2
      1.83970151   -0.91390364    1.10583534  1.00782503207d0    2
      1.58023648    2.10625129    1.27164940 1.00782503207d0     3
      2.14464785    1.23080044   -0.12912254 1.00782503207d0     3
      4.03977823    0.31375819    1.20306619 1.00782503207d0     4
      3.48797307    1.19751170    2.59554571 1.00782503207d0     4
      3.84329672    3.35364351    1.37650469 1.00782503207d0     5
      4.39964721    2.46369549   -0.02679096 1.00782503207d0     5
1 2 0 0 0    !  connectivity of frame chain
1 2 3 0 0    ! connectivity of frame chain
1 2 3 4 0    ! connectivity of frame chain
1 2 3 4 5    ! connectivity of frame chain
2 3          !start rotation axis
3 4
4 5
5 6
```

Lines 2–16 give the Cartesian coordinates (in Å) in the first three columns for each atom, the mass of each atom in the fourth column, and the label of the frame in the fifth column. Lines 17–20 give the connectivity of the frames that is explained in the next paragraph. Lines 21–24 give the rotation axis for each torsion; the numbers in these lines are the numbers of the atoms, not the frame numbers.

We use 1-butanol as an example to explain how to define the frames and their connectivity.

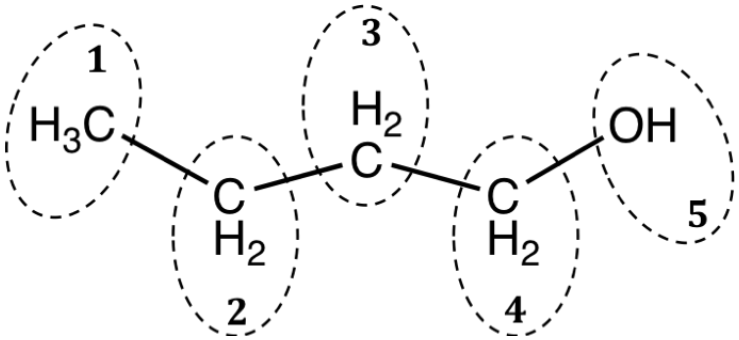


Figure 7-1 Definition of frames for 1-butanol. The group of atoms in each dash line circle is defined as a frame.

If frames are defined as Fig. 7-1, we can define the connectivity of frame chain as

```
1  2
1  2  3
1  2  3  4
1  2  3  4  5
```

Each line begins with frame 1 and ends with the destination frame, and includes all intervening frames. One can number the frames and specify their connectivity in various ways.

7. 4. Input files for the *mcvorm.exe* executable

The executable *mcvorm.exe* calculates $M_{j,\tau}$ values defined by Voronoi tessellation of the torsional space by Monte Carlo sampling. This program can handle cases of arbitrary dimensionality.

The input file for *mcvorm.exe* is illustrated by the following example.

Example:

```
 2        ! number of torsions (dimensions)
 5        !number of structures in the input
 0.005    !Monte Carlo standard error
 2  1     !symmetry number for each torsion
-152.3      65.1   ! values of the dihedral angles for the first structure
 159.7     178.8   ! values of the dihedral angles for the second structure
  84.3    -180.0   ! values of the dihedral angles for the third structure
 152.3     -65.1   ! values of the dihedral angles for the fourth structure
-159.7    -178.8   ! values of the dihedral angles for the fifth structure
```

The text after the ! symbol is an optional comment that explains the input of that line. Starting from the fifth line, each line lists the dihedral angles of the torsions for each structure.

7. 5. Input files for the *msinput.exe* executable

The *msinput.exe* is a utility program to generate a template main input file (named *mstor.inp*) and *hess.dat* input file by extracting the needed information from *Gaussian* formatted checkpoint files. All the *Gaussian* formatted checkpoint files must be combined into a single file and this single file is the input file of the *msinput.exe* program.

Note that the generated *mstor.inp* file cannot be used directly as an input file of the *mstor.exe* program because many parameters are not assigned and some of them are assigned to some temporary numbers (e.g., the non-redundant internal coordinates are not written). The Cartesian coordinates (in Å) and relative energies (in kcal/mol with the lowest energy set to 0) in the *mstor.inp* file and the Hessians in the *hess.dat* file are taken from the checkpoint file.

If the output *mvorm.out* file from *mcvorm.exe* is given, this utility code will take the M_j values from the *mvorm.out* file (do not rename the *mvorm.out* file name). However, if the Voronoi calculation only includes part of torsions, which is often the case, user needs to input other M values for uncoupled torsions manually in the *mstor.inp* file.

This program uses the *symmetry.exe* to determine the weight and the symmetry number of overall rotation of each structure.

7. 6. Input files for the *mvinput.exe* executable

The *mvinput.exe* is a utility program to generate a template input file, named as *mvorm.inp*, for the *vorm.exe* and/or *mcvorm.exe* programs by extracting the information from *Gaussian* checkpoint files. The user must combine all the formatted checkpoint files into a single file and write the number of torsions in the first line followed by numbers that label the four atoms that define each dihedral angle in successive lines.

The input file for *mvinput.exe* is illustrated by the following example:

Example:

```
2      ! number of torsions
1 5 7 10    !  the four atom numbers which define the first dihedral angle
5 7 14 15  !  the four atom numbers which define the second dihedral angle
… here begins the Gaussian formatted checkpoint file…
```

This programs uses the *symmetry.exe* to determine the symmetry point group of each structure. If the molecule has no symmetry (e.g., it has the symmetry of the C_1 point group) the values of the dihedral angles corresponding to its enantiomer are included. It

should be noted that the mirror images must be included only if the enantiomer interconversion between them is possible by internal rotation.

In all the cases the generated file must be checked carefully by the user before using it. For example, the symmetry numbers for all torsions in the generated file are always set to 1.

7. 7. Input files for the *tes.exe* executable

The *tes.exe* executable is a utility program to calculate 1-D torsional partition functions using the torsional eigenvalue summation (TES) method. The TES method is described in: B. A. Ellingson, V. A. Lynch, S. L. Mielke, and D. G. Truhlar, Journal of Chemical Physics **125**, 84305/1–17 (2006). Here is a brief description of the method.

A Fourier cosine series is used to represent a 1-D torsional potential

$$V = \sum_{j=0}^{j_{\max}} b_j \cos(j\phi) \qquad (44)$$

where $j_{\max}$ is the maximum order to be used, b_j is a coefficient, and ϕ is torsional coordinate. The user needs to input a reduced moment of inertia and a number of potential energy values along the torsional coordinate; these values are used to fit the coefficients of the Fourier cosine series. The Hamiltonian with this fitted potential is represented in a basis of the form $(1/\sqrt{2\pi})\exp(ik\phi)$ and diagonalized. The resulting eigenvalues are then used to calculate a partition function for a one-dimensional separable torsion with a constant reduced moment of inertia. This algorithm is not used anywhere in a MS-T calculation; it is simply provided in case a user wishes to calculate comparison results.

The input file for the *tes.exe* is illustrated by the following example.

Example:

```
2
0.7456101 ! moment of inertia (in amu*Å2)
200       ! number of eigenvalues to be calculated
1.00      ! frequency scaling factor
10        ! number of terms in a Fourier cosine series to be used in potential
fitting
41        ! number of input data (angles in degree and energy in kcal/mol)
59.65955        0      ! value of the angle in degree and energy in kcal/mol
68.65955        0.053651993
77.65955        0.206199357
    :                   :
3             ! number of temperatures
```

```
200         ! temperatures in K
800
2000
```

7. 8. Input files for the *vorm.exe* executable

The executable *vorm.exe* calculates M_j values using Voronoi tessellation and it can only handle 2-D and 3-D cases. This program calls *hull.exe* to perform the Voronoi tessellation calculations, and then reads the output of *hull.exe*.

The input file for the *vorm.exe* is illustrated by the following example.

Example:

```
 2         ! number of torsions
 5         ! number of structures
 2  1      ! symmetry number for each torsion
-152.3        65.1  ! values of the dihedral angles for the first structure
 159.7       178.8
  84.3      -180.0
 152.3       -65.1
-159.7      -178.8
```

Explanation of the example: The first line is the number of torsions; the second line gives the number of structures; the third line gives the symmetry number of each torsion; starting from the fourth line, each line has the dihedral angles of the torsions for each structure.

In the *mstor/utili/vorm-test/* directory, five examples of using *vorm.exe* for calculating M_j values are provided, and their output files are located in the *mstor/utili/vorm-test/* directory.

8. Test suites

The test suites have been designed to give some examples of input and output files for the *MSTor* program and its utility programs. The test suites include examples of radicals and transition states as well as closed-shell molecules. The test suite is located in the *mstor/testrun/* directory, and the output files are in the *mstor/testo/* directory. There are seven directories under *mstor/testrun/* as shown below:

Directory name	For program:
ConfGen-test	*ConfGen.exe*
kpmoments-test	*kpmoments.exe*
mcvorm-test	*mcvorm.exe*
msinput-test	*msinput.exe*
mstor-test	*mstor.exe*
mvinput-test	*mvinput.exe*
tes-test	*tes.exe*
vorm-test	*vorm.exe*

8. 1. Test runs for *ConfGen.exe*

One input file, 2hexyl.inp, for ConfGen.exe is given in the *mstor/testrun/ConfGen-test/* directory. This test run generates conformational structures for 2-hexyl radical. The initial structure is the all-trans structure. The ConfGen.exe program will generate 27 structures (including the initial one) by rotating three C–C bonds.

8. 2. Test runs for *kpmoments.exe*

Two input files, *kp_butanol.inp* and *kp_butanol.inp*, are given in the *mstor/testrun/kpmoments-test/* directory. The two input files use the same molecule/geometry, but they label the frames differently.

8. 3. Test runs for *mcvorm.exe*

Seven input files for the *mcvorm.exe* are given in the *mstor/testrun/mcvorm-test/* directory. They cover cases from 2D to 6D Monte Carlo sampling of the Voronoi volumes. All test runs set the target Monte Carlo standard error for *M* values to 0.005. These test runs are outlined below.

Dimensions	Input files	Description
2D	pentyl-2d-set1.dat	5 structures of 1-pentyl radical using a $NS:SC=2:2$ scheme
	pentyl-2d-set2.dat	5 structures of 1-pentyl radical using $NS:SC=2:2$.
3D	butanol.dat	29 structures of 1-butanol
	pentyl_3d.dat	15 structures of 1-pentyl radical
4D	test-4d.dat	16 evenly distributed points in 4D torsional space with each torsional coordinate going from 0 to 360 degrees.
5D	test-5d.dat	32 evenly distributed points in 5D torsional space with each torsional coordinate going from 0 to 360 degrees.
6D	test-6d.dat	64 evenly distributed points in 6D torsional space with each torsional coordinate going from 0 to 360 degrees.

8. 4. Test runs for *msinput.exe*

One input file (*ethanol-all.fchk*) is in the *mstor/testrun/msinput-test/* directory. This input file contains two formatted checkpoint files of ethanol (two conformational structures) from *Gaussian 09* calculations. The msinput.exe program will generate two output files, *mstor.inp* and *hess.dat* from this run.

8. 5. Test runs for *mvinput.exe*

One input file (*1-pentyl-all.fchk*) is in the *mstor/testrun/mvinput-test/* directory. This input file contains eight formatted checkpoint files of 1-pentyl radical (eight conformational structures) from *Gaussian 09* calculations. The *mvinput.exe* program will generate one output file, *mvorm.inp* from this run. This *mvorm.inp* file can be used as a template of the input file for *vorm.exe*/*mcvorm.exe* calculation.

8. 6. Test runs for *mstor.exe*

8. 6. 1. Ethanol

Input files for ethanol are given in the *mstor/testrun/mstor-test/ethanol/* directory.

8. 6. 2. 1-Butanol

Input files for 1-butanol are given in the *mstor/testrun/mstor-test/butanol/* directory.

8. 6. 3. 1-Pentyl radical

Input files for 1-pentyl are given in the *mstor/testrun/mstor-test/pentyl/* directory. Three test runs using different NS:SC schemes for $M_{j,\tau}$ values are given in this directory. They all share the same *hess.dat* file for Hessians.

Test run main input	Description
1-pentyl-intM.dat	Integer $M_{j,\tau}$ values (4:0 scheme)
1-pentyl-2dvor.dat	$M_{j,\tau}$ values obtained by $\mathrm{NS:SC} = 2:2$
1-pentyl-3dvor.dat	$M_{j,\tau}$ values obtained by $\mathrm{NS:SC} = 1:3$.

8. 6. 4. 1,4-Hydrogen shift saddle point of pentyl radical

Input files for 1, 4-hydrogen shift saddle point of pentyl radical are given in the *mstor/testrun/mstor-test/pentyl-ts14/* directory. There are two pairs of mirror images for the saddle point in the input file, but only the terminal methyl group torsion is treated by torsional corrections because these four structures cannot be interconverted through internal rotations.

8. 7. Test runs for *tes.exe*

One input file, *ethanol-OH.inp*, is given in the *mstor/testrun/tes-test/* directory. This input file contains a potential of the internal rotation of the hydroxyl group in ethanol.

8. 8. Test runs for *vorm.exe*

Five input files are given in the *mstor/testrun/vorm-test/* directory for the *vorm.exe* program. These test runs are described below.

Dimensions	Input files	Description
2D	pentyl-2d-set1.dat	5 structures of 1-pentyl radical using $\mathrm{NS:SC} = 2:2$
	pentyl-2d-set2.dat	5 structures of 1-pentyl radical using $\mathrm{NS:SC} = 2:2$
	pentyl-2d-set3.dat	5 structures of 1-pentyl radical using $\mathrm{NS:SC} = 2:2$
3D	butanol.dat	29 structures of 1-butanol
	pentyl_3d.dat	15 structures of 1-pentyl radical using $\mathrm{NS:SC} = 1:3$

9. Computers, operating systems, and compilers on which the code has been tested

9.1. Version 2011

Computer	OS	Compiler[1]
Itasca[2]	SuSE Linux Enterprise Server 11 SP1	ifort/11.1 and icc/11.1 gfortran/4.4 and gcc 4.4
Calhoun[3]	SuSE Linux Enterprise Server 10 SP3	ifort/11.1 and icc/11.1 gfortran/4.4 and gcc 4.4
Koronis[4]	SuSE Linux Enterprise Server 11 SP1	ifort/11.1 and icc/11.1 gfortran/4.4.3 and gcc/4.4.3
Elmo[5]	SuSE Linux Enterprise Server 10 SP3	ifort/11.1 and icc/11.1 gfortran/4.4.3 and gcc/4.4.3
Mac (Intel Xeon)	Mac OS X 10.5.8	gfortran/4.4 and gcc 4.4

1. On most computers two sets of compilers were tested.
2. Itasca is a Linux cluster of HP Proliant BL280c G6 Blade servers.
3. Calhoun is an SGI Altix XE 1300 cluster.
4. Koronis is a constellation of SGI systems.
5. Elmo is a Sun Fire X4600 Linux cluster.

9.2. Version 2011-2

Computer	OS	Compiler[1]
Itasca[2]	SuSE Linux Enterprise Server 11 SP1	ifort/11.1 and icc/11.1 gfortran/4.4 and gcc 4.4
Calhoun[3]	SuSE Linux Enterprise Server 10 SP3	ifort/11.1 and icc/11.1 gfortran/4.4 and gcc 4.4
Koronis[4]	SuSE Linux Enterprise Server 11 SP1	ifort/11.1 and icc/11.1 gfortran/4.4.3 and gcc/4.4.3
Elmo[5]	SuSE Linux Enterprise Server 10 SP3	ifort/11.1 and icc/11.1 gfortran/4.4.3 and gcc/4.4.3
Mac (Intel Xeon)	Mac OS X 10.5.8	gfortran/4.4 and gcc 4.4

1. On most computers two sets of compilers were tested.
2. Itasca is a Linux cluster of HP Proliant BL280c G6 Blade servers.

3. Calhoun is an SGI Altix XE 1300 cluster.
4. Koronis is a constellation of SGI systems.
5. Elmo is a Sun Fire X4600 Linux cluster.

9.3. Version 2011-3

Computer	OS	Compiler[1]
Itasca[2]	SuSE Linux Enterprise Server 11 SP1	ifort/11.1 and icc/11.1 gfortran/4.4 and gcc 4.4
Calhoun[3]	SuSE Linux Enterprise Server 10 SP3	ifort/11.1 and icc/11.1 gfortran/4.4 and gcc 4.4
Koronis[4]	SuSE Linux Enterprise Server 11 SP1	ifort/11.1 and icc/11.1 gfortran/4.4.3 and gcc/4.4.3
Elmo[5]	SuSE Linux Enterprise Server 10 SP3	ifort/11.1 and icc/11.1 gfortran/4.4.3 and gcc/4.4.3
Mac (Intel Xeon)	Mac OS X 10.5.8	gfortran/4.4 and gcc 4.4

1. On most computers two sets of compilers were tested.
2. Itasca is a Linux cluster of HP Proliant BL280c G6 Blade servers.
3. Calhoun is an SGI Altix XE 1300 cluster.
4. Koronis is a constellation of SGI systems.
5. Elmo is a Sun Fire X4600 Linux cluster.

9.4. Version 2013

Computer	OS	Compiler[1]
Itasca[2]	CentOS 6.2	ifort/11.1 and icc/11.1 gfortran/4.4 and gcc 4.4
Calhoun[3]	CentOS 6.2	ifort/11.1 and icc/11.1 gfortran/4.4 and gcc 4.4
Koronis[4]	SuSE Linux Enterprise Server 11 SP1	ifort/11.1 and icc/11.1 gfortran/4.4.3 and gcc/4.4.3 gfortran/4.4.3 and gcc/4.4.3
Mac (Intel Xeon)	Mac OS X 10.8.2	gfortran/4.4 and gcc 4.4

1. On most computers two sets of compilers were tested.
2. Itasca is a Linux cluster of HP Proliant BL280c G6 Blade servers.
3. Calhoun is an SGI Altix XE 1300 cluster.

4. Koronis is a constellation of SGI systems.

10. Acknowledgments

We are grateful to Serguei Patchkovskii for providing the symmetry code. This work was supported in part by the U.S. Department of Energy (DOE) Office of Science, Office of Basic Energy Sciences, as part of the Combustion Energy Frontier Research Center under Award Number DE-SC0001198. This work was also supported by the DOE through grant no. DE-FG02-86ER13579.

11. Revision history

Version 2011 (October 6, 2011)

Authors: J. Zheng, S. L. Mielke, K. L. Clarkson, and D. G. Truhlar

This is the first distributed version.

Version 2011-2 (November 2, 2011)

Authors: J. Zheng, S. L. Mielke, K. L. Clarkson, and D. G. Truhlar

This revision adds the capability to calculate the heat capacity at constant pressure by using a four-point central finite difference scheme.

In addition, more comments were added to the source code.

Version 2011-3 (February 15, 2011)

Authors: J. Zheng, S. L. Mielke, K. L. Clarkson, and D. G. Truhlar

This is the first version distributed by *CPC*.

This revision adds one more utility programs, in particular a program to calculate 1-D torsional partition functions using the torsional eigenvalue summation (TES) method.

Version 2011-3-A (December 19, 2012)

Authors: J. Zheng, S. L. Mielke, K. L. Clarkson, and D. G. Truhlar
Contributors to this revision: J. Zheng and D. G. Truhlar

This revision has a bug fixed in the vib.f90 file. Line 33 is uncommented to initialize the array of center-of-mass coordinates as zero.

Version 2013 (January 31, 2013)

Authors: J. Zheng, S. L. Mielke, K. L. Clarkson, R. Meana-Pañeda, and D. G. Truhlar
Contributors to new revision: J. Zheng, R. Meana-Pañeda, and D. G. Truhlar

This revision is submitted to *Computer Physics Communication* in February 2013 by means of a new version announcement.

New features and bug fixes:

- The MS-T method based on a coupled torsional potential is added, which is called MS-T(C), where C denotes coupled. The original method that is based on an uncoupled potential is called MS-T(U).

- The capability of treating linear bending motions is added.

- The sections $framechain and $framedef are no longer needed in the input file, which simplifies the input file.

- The code for evaluating of the point group symmetry of a structure from Serguei Patchkovskii is included.

- The *mvinput.exe* utility program is added to generate the input file for the *vorm.exe* and/or *mcvorm.exe* codes. The *msinput.exe* utility code has been modified to write the *M* values taken from the *mvorm.out* file. Both *mvinput.exe* and *msinput.exe* use the *symmetry.exe* program to determine the point group of each structure.

- The *ConfGen.exe* has been completely written in Fortran 90 and reads news fields corresponding to the Gaussian input keywords.

- A bug in the generation of different structures that affects *ConfGen.exe*, *vorm.exe* and *mcvorm.exe* utility codes has been fixed