|  |  |
| --- | --- |
| **1.** | **Potential Energy Programs** |

This section of the manual contains a brief description of each of the TINKER potential energy programs. A detailed example showing how to run each program is included in a later section. The programs listed below are all part of the main, supported distribution. Additional source code for various unsupported programs can be found in the /other directory of the TINKER distribution.

**ALCHEMY**

A simple program to perform very basic free energy perturbation calculations. This program is provided mostly for demonstration purposes. For example, we use ALCHEMY in a molecular modeling course laboratory exercise to perform such classic mutations as chloride to bromide and ethane to methanol in water. The present version uses the perturbation formula and windowing with an explicit mapping of atoms involved in the mutation (``AMBER''-style), instead of thermodynamic integration and independent freely propagating groups of mutated atoms (``CHARMM''-style). Some of the code specific to this program is limited to the AMBER and OPLS potential functional forms, but could be easily generalized to handle other potentials. A more general and sophisticated version is currently under development.

**ANALYZE**

Provides information about a specific molecular structure. The program will ask for the name of a structure file, which must be in the TINKER **.xyz** file format, and the type of analysis desired. Options allow output of: (1) total potential energy of the system, (2) breakdown of the energy by potential function type or over individual atoms, (3) computation of the total dipole moment and its components, moments of inertia and radius of gyration, (4) listing of the parameters used to compute selected interaction energies, (5) energies associated with specified individual interactions.

**ANNEAL**

Performs a molecular dynamics simulated annealing computation. The program starts from a specified input molecular structure in TINKER **.xyz** format. The trajectory is updated using either a modified Beeman or a velocity Verlet integration method. The annealing protocol is implemented by allowing smooth changes between starting and final values of the system temperature via the Groningen method of coupling to an external bath. The scaling can be linear or sigmoidal in nature. In addition, parameters such as cutoff distance can be transformed along with the temperature. The user must input the desired number of dynamics steps for both the equilibration and cooling phases, a time interval for the dynamics steps, and an interval between coordinate/trajectory saves. All saved coordinate sets along the trajectory are placed in sequentially numbered cycle files.

**DYNAMIC**

Performs a molecular dynamics (MD) or stochastic dynamics (SD) computation. Starts either from a specified input molecular structure (an **.xyz** file) or from a structure-velocity-acceleration set saved from a previous dynamics trajectory (a restart from a **.dyn** file). MD trajectories are propagated using either a modified Beeman or a velocity Verlet integration method. SD is implemented via our own derivation of a velocity Verlet-based algorithm. In addition the program can perform full crystal calculations, and can operate in constant energy mode or with maintenance of a desired temperature and/or pressure using the Groningen method of coupling to external baths. The user must input the desired number of dynamics steps, a time interval for the dynamics steps, and an interval between coordinate/trajectory saves. Coordinate sets along the trajectory can be saved as sequentially numbered cycle files or directly to a TINKER archive **.arc** file. At the same time that a point along the trajectory is saved, the complete information needed to restart the trajectory from that point is updated and stored in the **.dyn** file.

**GDA**

A program to implement Straub's Gaussian Density Annealing algorithm over an effective series of analytically smoothed potential energy surfaces. This method can be viewed as an extended stochastic version of the diffusion equation method of Scheraga, *et al*., and also has many similar features to the TINKER Potential Smoothing and Search (PSS) series of programs. The current version of GDA is similar to but does not exactly reproduce Straub's published method and is limited to argon clusters and other simple systems involving only van der Waals interactions; further modification and development of this code is currently underway in the Ponder research group. As with other programs involving potential smoothing, GDA currently requires use of the **smooth.prm** force field parameters.

**MINIMIZE**

The MINIMIZE program performs a limited memory L-BFGS minimization of an input structure over Cartesian coordinates using a modified version of the algorithm of Jorge Nocedal. The method requires only the potential energy and gradient at each step along the minimization pathway. It requires storage space proportional to the number of atoms in the structure. The MINIMIZE procedure is recommended for preliminary minimization of trial structures to an *rms* gradient of 1.0 to 0.1 kcal/mole/≈. It has a relatively fast cycle time and is tolerant of poor initial structures, but converges in a slow, linear fashion near the minimum. The user supplies the name of the TINKER **.xyz** coordinates file and a target *rms* gradient value at which the minimization will terminate. Output consists of minimization statistics written to the screen or redirected to an output file, and the new coordinates written to updated **.xyz** files or to cycle files.

**MINIROT**

The MINIROT program uses the same limited memory L-BFGS method as MINIMIZE, but performs the computation in terms of dihedral angles instead of Cartesian coordinates. Output is saved in an updated **.int** file or in cycle files.

**MINRIGID**

The MINRIGID program is similar to MINIMIZE except that it operates on rigid bodies starting from a TINKER **.xyz** coordinate file and the rigid body group definitions found in the corresponding **.key** file. Output is saved in an updated **.xyz** file or in cycle files.

**MONTE**

The MONTE program implements the Monte Carlo Minimization algorithm developed by Harold Scheraga's group and others. The procedure takes Monte Carlo steps for either a single atom or a single torsional angle, then performs a minimization before application of the Metropolis sampling method. This results in effective sampling of a modified potential surface where the only possible energy levels are those of local minima on the original surface. The program can be easily modified to elaborate on the available move set.

**NEWTON**

A truncated Newton minimization method which requires potential energy, gradient and Hessian information. This procedure has significant advantages over standard Newton methods, and is able to minimize very large structures completely. Several options are provided with respect to minimization method and preconditioning of the Newton equations. The default options are recommended unless the user is familiar with the math involved. This program operates in Cartesian coordinate space and is fairly tolerant of poor input structures. Typical algorithm iteration times are longer than with nonlinear conjugate gradient or variable metric methods, but many fewer iterations are required for complete minimization. NEWTON is usually the best choice for minimizations to the 0.01 to 0.000001 kcal/mole/≈ level of *rms* gradient convergence. Tests for directions of negative curvature can be removed, allowing NEWTON to be used for optimization to conformational transition state structures (this only works if the starting point is very close to the transition state). Input consists of a TINKER **.xyz** coordinates file; output is an updated set of minimized coordinates and minimization statistics.

**NEWTROT**

The NEWTROT program is similar to NEWTON except that it requires a **.int** file as input and then operates in terms of dihedral angles as the minimization variables. Since the dihedral space Hessian matrix of an arbitrary structure is often indefinite, this method will often not perform as well as the other, simpler dihedral angle based minimizers.

**OPTIMIZE**

The OPTIMIZE program performs a optimally conditioned variable metric minimization of an input structure over Cartesian coordinates using an algorithm due to William Davidon. The method does not perform line searches, but requires computation of energies and gradients as well as storage for an estimate of the inverse Hessian matrix. The program operates on Cartesian coordinates from a TINKER **.xyz** file. OPTIMIZE will typically converge somewhat faster and more completely than MINIMIZE. However, the need to store and manipulate a full inverse Hessian estimate limits its use to structures containing less than a few hundred atoms on workstation class machines. As with the other minimizers, OPTIMIZE needs input coordinates and an *rms* gradient cutoff criterion. The output coordinates are saved in updated **.xyz** files or as cycle files.

**OPTIROT**

The OPTIROT program is similar to OPTIMIZE except that it operates on dihedral angles starting from a TINKER **.int** internal coordinate file. This program is usually the preferred method for most dihedral angle optimization problems since Truncated Newton methods appear, in our hands, to lose some of their efficacy in moving from Cartesian to torsional coordinates.

**OPTRIGID**

The OPTRIGID program is similar to OPTIMIZE except that it operates on rigid bodies starting from a TINKER **.xyz** coordinate file and the rigid body atom group definitions found in the corresponding **.key** file. Output is saved in an updated **.xyz** file or in cycle files.

**PATH**

A program that implements a variant of Elber's Lagrangian multiplier-based reaction path following algorithm. The program takes as input a pair of structural minima as TINKER **.xyz** files, and then generates a user specified number of points along a path through conformational space connecting the input structures. The intermediate structures are output as TINKER cycle files, and the higher energy intermediates can be used as input to a Newton-based optimization to locate conformational transition states.

**PSS**

Implements our version of a potential smoothing and search algorithm for the global optimization of molecular conformation. An initial structure in **.xyz** format is first minimized in Cartesian coordinates on a series of increasingly smoothed potential energy surfaces. Then the smoothing procedure is reversed with minimization on each successive surface starting from the coordinates of the minimum on the previous surface. A local search procedure is used during the backtracking to explore for alternative minima better than the one found during the current minimization. The final result is usually a very low energy conformation or, in favorable cases, the global energy minimum conformation. The minimum energy coordinate sets found on each surface during both the forward smoothing and backtracking procedures are placed in sequentially numbered cycle files.

**PSSRIGID**

This program implements the potential smoothing and search method as described above for the PSS program, but performs the computation in terms of keyfile-defined rigid body atom groups instead of Cartesian coordinates. Output is saved in numbered cycle files with the **.xyz** file format.

**PSSROT**

This program implements the potential smoothing and search method as described above for the PSS program, but performs the computation in terms of a set of user-specified dihedral angles instead of Cartesian coordinates. Output is saved in numbered cycle files with the **.int** file format.

**SADDLE**

A program for the location of a conformational transition state between two potential energy minima. SADDLE uses a conglomeration of ideas from the Bell-Crighton quadratic path and the Halgren-Lipscomb synchronous transit methods. The basic idea is to perform a nonlinear conjugate gradient optimization in a subspace orthogonal to a suitably defined reaction coordinate. The program requires as input the coordinates (TINKER **.xyz** files) of the two minima and an *rms* gradient convergence criterion for the optimization. The current estimate of the transition state structure is written to the file TSTATE.XYZ. Crude transition state structures generated by SADDLE can sometimes be refined using the NEWTON program. Optionally, a scan of the interconversion pathway can be made at each major iteration.

**SCAN**

A program for general conformational search of an entire potential energy surface via a basin hopping method. The program takes as input a TINKER **.xyz** coordinates file which is then minimized to find the first local minimum for a search list. A series of activations along various normal modes from this initial minimum are used as seed points for additional minimizations. Whenever a previously unknown local minimum is located it is added to the search list. When all minima on the search list have been subjected to the normal mode activation without locating additional new minima, the program terminates. The individual local minima are written to cycle files as they are discovered. While the SCAN program can be used on standard undeformed potential energy surfaces, we have found it to be most useful for quickly ``scanning'' a smoothed energy surface to enumerate the major basins of attraction spaning the entire surface.

**SNIFFER**

A program that implements the Sniffer global optimization algorithm of Butler and Slaminka, a discrete version of Griewank's global search trajectory method. The program takes an input TINKER **.xyz** coordinates file and shakes it vigorously via a modified dynamics trajectory before, hopefully, settling into a low lying minimum. Some trial and error is often required as the current implementation is sensitive to various parameters and tolerances that govern the computation. At present, these parameters are not user accessible, and must be altered in the source code. However, this method can do a good job of quickly optimizing conformation within a limited range of convergence.

**TESTGRAD**

The TESTGRAD program computes and compares the analytical and numerical first derivatives (*i.e.*, the gradient vector) of the potential energy for a Cartesian coordinate input structure. The output can be used to test or debug the current potential or any added user defined energy terms.

**TESTHESS**

The TESTHESS program computes and compares the analytical and numerical second derivatives (*i.e.*, the Hessian matrix) of the potential energy for a Cartesian coordinate input structure. The output can be used to test or debug the current potential or any added user defined energy terms.

**TESTLIGHT**

A program to compare the efficiency of different nonbonded neighbor methods for the current molecular system. The program times the computation of energy and gradient for the van der Waals and charge-charge electrostatic potential terms using a simple double loop over all interactions and using the Method of Lights algorithm to select neighbors. The results can be used to decide whether the Method of Lights has any CPU time advantage for the current structure. Both methods should give exactly the same answer in all cases, since the identical individual interactions are computed by both methods. The default double loop method is faster when cutoffs are not used, or when the cutoff sphere contains about half or more of the total system of unit cell. In cases where the cutoff sphere is much smaller than the system size, the Method of Lights can be much faster since it avoids unnecessary calculation of distances beyond the cutoff range.

**TESTROT**

The TESTROT program computes and compares the analytical and numerical first derivatives (*i.e.*, the gradient vector) of the potential energy with respect to dihedral angles. Input is a TINKER **.int** internal coordinate file. The output can be used to test or debug the current potential functions or any added user defined energy terms.

**TIMER**

A simple program to provide timing statistics for energy function calls within the TINKER package. TIMER requires an input **.xyz** file and outputs the CPU time (wall clock time on some machine types) needed to perform a specified number of energy, gradient and Hessian evaluations.

**TIMEROT**

This program is similar to TIMER, only it operates over dihedral angles via input of a TINKER **.int** internal coordinate file. In the current version, the torsional Hessian is computed numerically from the analytical torsional gradient.

**VIBRATE**

A program to perform vibrational analysis by computing and diagonalizing the full Hessian matrix (*i.e.*, the second partial derivatives) for an input structure (a TINKER **.xyz** file). Eigenvalues and eigenvectors of the mass weighted Hessian (*i.e.*, the vibrational frequencies and normal modes) are also calculated. Structures corresponding to individual normal mode motions can be saved in cycle files.

**VIBROT**

The program VIBROT forms the torsional Hessian matrix via numerical differentiation of the analytical torsional gradient. The Hessian is then diagonalized and the eigenvalues are output. The present version does not compute the kinetic energy matrix elements needed to convert the Hessian into the torsional normal modes; this will be added in a later version. The required input is a TINKER **.int** internal coordinate file.

**XTALFIT**

The XTALFIT program is of use in the automated fitting of potential parameters to crystal structure and thermodynamic data. XTALFIT takes as input several crystal structures (TINKER **.xyz** files with unit cell parameters in corresponding keyfiles) as well as information on lattice energies and dipole moments of monomers. The current version uses a nonlinear least squares optimization to fit van der Waals and electrostatic parameters to the input data. Bounds can be placed on the values of the optimization parameters.

**XTALMIN**

A program to perform full crystal minimizations. The program takes as input the structure coordinates and unit cell lattice parameters. It then alternates cycles of Newton-style optimization of the structure and conjugate gradient optimization of the crystal lattice parameters. This alternating minimization is slower than more direct optimization of all parameters at once, but is somewhat more robust in our hands. The symmetry of the original crystal is not enforced, so interconversion of crystal forms may be observed in some cases.