MacroModel 9.7

Reference Manual

Schrödinger Press

MacroModel Reference Manual Copyright © 2009 Schrödinger, LLC. All rights reserved.

While care has been taken in the preparation of this publication, Schrödinger assumes no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

Canvas, CombiGlide, ConfGen, Epik, Glide, Impact, Jaguar, Liaison, LigPrep, Maestro, Phase, Prime, PrimeX, QikProp, QikFit, QikSim, QSite, SiteMap, Strike, and WaterMap are trademarks of Schrödinger, LLC. Schrödinger and MacroModel are registered trademarks of Schrödinger, LLC. MCPRO is a trademark of William L. Jorgensen. Desmond is a trademark of D. E. Shaw Research. Desmond is used with the permission of D. E. Shaw Research. All rights reserved. This publication may contain the trademarks of other companies.

Schrödinger software includes software and libraries provided by third parties. For details of the copyrights, and terms and conditions associated with such included third party software, see the Legal Notices for Third-Party Software in your product installation at $SCHRODINGER/docs/html/third_party_legal.html (Linux OS) or %SCHRODINGER%\docs\html\third_party_legal.html (Windows OS).

This publication may refer to other third party software not included in or with Schrödinger software ("such other third party software"), and provide links to third party Web sites ("linked sites"). References to such other third party software or linked sites do not constitute an endorsement by Schrödinger, LLC. Use of such other third party software and linked sites may be subject to third party license agreements and fees. Schrödinger, LLC and its affiliates have no responsibility or liability, directly or indirectly, for such other third party software and linked sites, or for damage resulting from the use thereof. Any warranties that we make regarding Schrödinger products and services do not apply to such other third party software or linked sites, or to the interaction between, or interoperability of, Schrödinger products and services and such other third party software.

June 2009

Contents

Document Conventions vii

Chapter 1: Capabilities and New Features 1

1.1 General Description 1

1.2 Energetic Calculations 1

1.2.1 Force Fields and Energy Equations 1

1.2.2 Energy Minimization Methods 4

1.2.3 Conformational Searching and Comparison 5

1.2.4 Molecular Dynamics and Monte Carlo Simulations 6

1.2.5 Coordinate Manipulation 6

1.2.6 Molecular Association 6

1.2.7 Partition Coefficients 7

1.2.8 Free-energy Perturbation 7

1.2.9 Monte Carlo Simulation 7

1.3 Computational Features 7

1.4 Program Capacity 9

1.5 Miscellaneous 9

1.6 Citing MacroModel in Publications 9

Chapter 2: Running MacroModel 11

2.1 Running MacroModel From the Command Line 11

2.1.1 Environment Variables 11

2.1.2 Command-line Syntax 12

2.1.3 Storage of Temporary Files 13

2.1.4 The .com File 14

2.1.4.1 File Specifications 14

2.1.4.2 Operation Code Specifications 15

2.1.4.3 Using an Executable .com File 16

2.2 Job Control 17

2.2.1 Interacting With a Running MacroModel Job Via Files 17

2.2.1.1 Stopping MacroModel 18

2.2.1.2 Putting MacroModel to Sleep 18
2.2.1.3 Triggering an Interim Job Summary 19

2.3 Network-Distributed Processing 19

2.3.1 Load Balancing 20
2.3.2 Types of Distributed Processing 20
2.3.3 MacroModel-Based Distributed Processing 20
2.3.3.1 Internal Distribution 21
2.3.3.2 External Distribution 23

Chapter 3: Running MacroModel with the Simplified Input File **27**

Chapter 4: Operation Codes **31**

4.1 File Reading and Writing 31

4.2 Selection of Force Field, Nonbonded Cutoffs, and Solvation Treatment 33

4.3 Program Flow Control 44

4.4 Hydrogen Addition and Deletion 45

4.5 Energy Calculation 45

4.6 Energy Minimization 52

4.7 Constrained Energy Minimization 63

4.8 Conformational Comparison 74

4.9 Coordinate Manipulation 81

4.10 Conformational Searching 83

4.11 Molecular Dynamics and Monte Carlo Simulations 117

4.12 Free-Energy Perturbation 135

4.13 Monte Carlo Simulation 137

4.14 Docking 138

4.15 Miscellaneous Commands and Debugging 141

4.16 ConfGen 155

Chapter 5: MINTA 165

5.1 Overview 165

5.2 Introduction to MINTA 166

5.3 MINTA Methodology 168

5.4 MINTA Commands 169

5.5 Notes 172

5.6 Working Examples 173

5.6.1 Cyclononane 173

5.6.2 Glucose 175

5.6.3 Vancomycin 176

Appendix A: MacroModel Test Suites 177

Appendix B: MacroModel Benchmark Suites 179

Appendix C: Atom and Bond Types 181

C.1 The atom.typ File 181

C.2 MacroModel Bond Types 181

C.3 MacroModel Atom Types 182

Appendix D: Force-Field File Format 189

D.1 Organization of the Force-Field File 189

D.2 Introductory Section 190

D.2.1 Equation Specification 191

D.2.2 Charge Processing 194

D.2.3 ALT and SEL Specification 195

D.2.4 Atom Type Equivalencies 197

D.3 Main Parameter Section 197

D.3.1 Interactions 197

D.3.2 Adding New Parameters 206

D.3.3 Special Notes for AMBER 207
D.3.4 Special Notes for MM2 and MM3 210

D.4 Substructure Section 211
D.4.1 Use of Substructures 211
D.4.2 Substructure Linear Notation 212
D.4.2.1 Ring Closure 212
D.4.2.2 Chain Branching 213
D.4.2.3 Optional Atoms 213
D.4.3 Substructure Charge Processing 214
D.4.4 Substructure Examples 214
D.4.5 Geometry-dependent Parameters 216

Appendix E: The BMFF Protocol **221**

E.1 Purpose 221

E.2 Overview 221

E.3 MMFF Considerations 223

Appendix F: Format of .grd Files **225**

Appendix G: MINTA Background **227**

G.1 Introduction 227

G.2 The Thermodynamic Cycle 228

G.3 Computational Methods 229

G.4 References 233

References **237**

Opcode Index **243**

Document Conventions

In addition to the use of italics for names of documents, the font conventions that are used in this document are summarized in the table below.

Font	Example	Use
Sans serif	Project Table	Names of GUI features, such as panels, menus, menu items, buttons, and labels
Monospace	`$SCHRODINGER/maestro`	File names, directory names, commands, environment variables, and screen output
Italic	*filename*	Text that the user must replace with a value
Sans serif uppercase	CTRL+H	Keyboard keys

Links to other locations in the current document or to other PDF documents are colored like this: Document Conventions.

In descriptions of command syntax, the following UNIX conventions are used: braces `{ }` enclose a choice of required items, square brackets `[ ]` enclose optional items, and the bar symbol | separates items in a list from which one item must be chosen. Lines of command syntax that wrap should be interpreted as a single command.

File name, path, and environment variable syntax is generally given with the UNIX conventions. To obtain the Windows conventions, replace the forward slash / with the backslash \ in path or directory names, and replace the $ at the beginning of an environment variable with a % at each end. For example, `$SCHRODINGER/maestro` becomes `%SCHRODINGER%\maestro`.

In this document, to *type* text means to type the required text in the specified location, and to *enter* text means to type the required text, then press the ENTER key.

References to literature sources are given in square brackets, like this: [10].

Capabilities and New Features

1.1 General Description

The original MacroModel graphical user interface (GUI) is no longer supported as of MacroModel 7.2. It was replaced by Maestro, the GUI for all Schrödinger's products. Therefore, the MacroModel name now refers both to what was formerly called the BatchMin executable, and to the software package containing that executable and the XCluster program.

MacroModel is the batch-mode molecular modeling computational facility for use with Maestro. It is designed to minimize the energy of one structure or a series of structures, to eliminate duplicate conformations, to do conformational searches, and to conduct molecular dynamics simulations, including free-energy perturbation methods and mixed-mode Monte Carlo/stochastic dynamics procedures. MacroModel jobs may be started from the Maestro GUI, or may be run from the command-line. Some MacroModel features are not available from the Maestro interface. This manual attempts a complete description of MacroModel's facilities.

1.2 Energetic Calculations

1.2.1 Force Fields and Energy Equations

- MM2*, MM3*, AMBER*, and OPLS* force fields (enhanced and/or modified versions of the corresponding native force-fields; see Appendix D for implementation differences).
- Native OPLS_2001[1] (formerly named OPLS-AA).
- Native OPLS_2005[1].
- Native MMFF94 and MMFF94s.
- Native Amber 4.1 (AMBER94).
- Automatic checking and adjustment of formal charges and bond orders in calculations involving MMFF, MMFFs, and OPLS-AA force fields. This feature is disabled by `DEBG` 191.

1. In some places in this manual, the OPLS_2001 and OPLS_2005 force fields will be referred to collectively as OPLS-AA.

- BMFF (BatchMin Force-Field inter-process-communication library) allows an external process to generate force-field parameters and pass them to MacroModel. See Appendix E for details. This mechanism was used to implement MMFF, using a co-process supplied by Dr. Thomas A. Halgren; however, the mechanism is general and should greatly simplify the porting of additional force fields to MacroModel.
- Analytical continuum treatment of solvation (water, chloroform, and octanol setup files supplied). For precise parameter matching, it is possible to specify atomic connectivity out to the beta-atom level in `.slv` files.
- Ability to specify multiple parameter blocks in the solvation files for use with different force fields. This is done by placing a force-field number on an `SMODEL` line of the file. If an `SMODEL` block is found containing both the force field and the solvation model specified in the `.com` file, that block is used. If no block is found specifying the current force field, then, if a block is found for the current model specifying 0 for the force field, this block is used as the default. If neither is found, the first `SMODEL` block for the solvation model specified in the `.com` file is used. The program can also be instructed to use the general set of parameters for a particular solvent using arg2 of `SOLV`. Parametrizations for specific force fields are provided for OPLS-AA and MMFF for water and octanol.
- GB/SA with fixed/frozen atoms. We have continued to improve the use of both the GB and the SA parts of GB/SA when fixed and/or frozen atoms are in use. The intent is to provide speed optimization with minimal loss of accuracy. For energetic calculations, frozen atoms need not "see" each other; thus, for example, frozen nonbonded pairs are deleted from the nonbonded pairlist. However, the motion of the moving atoms causes the GB radii of the frozen atoms to change, and the GB energy contributions of frozen atom pairs will then change accordingly. To accommodate this without losing all the advantage of frozen-atom optimization, starting with MacroModel 7.0 we define a cutoff, `CUTFMM`, which defines a radius around the moving atoms within which any frozen atoms are treated as moving for the purpose of GB only. This is specified in arg8 of the `EXNB` command. Similar concerns arise for SA, where a `CUTSFM` is defined in arg5 of `EXN2`, an extension of the `EXNB` command.
- Correction for Born radii when not all nonbonded pairs are included in a calculation. For calculations on large systems such as proteins, typically not all nonbonded interactions are included in energetic and derivative calculations, to make simulations tractable. This leads to systematic errors in the values of Born radii used to calculate the GB energy in the GB/SA solvation model. A correction is done to account for this error and improve energetic and derivative calculations. See Section 2.3 of the *MacroModel User Manual* for further details.
- Vacuum, constant dielectric and distance-dependent-dielectric solvation treatments.

- Novel method for accurate electrostatic and GB energies despite truncation of nonbonded interactions. Implementation of the "Bond Dipole Cutoff" (BDCO) algorithm allows for exclusion of distant pairs from the nonbonded pairlist without compromising the absolute accuracy of the electrostatic and GB energies of a system. Much better agreement is achieved with the values of these energetic terms as calculated without cutoffs when BDCO is used as opposed to traditional residue-based cutoffs. Additionally, BDCO offers much better energetic accuracy than residue-based cutoffs using default cutoff values despite the fact that the BDCO calculation uses significantly *fewer* nonbonded pairs. Performance is maintained while accuracy is significantly improved. See the BDCO opcode description in this manual and Section 2.4 of the *MacroModel User Manual* for details.
- Dihedral driving for production of Ramachandran-style energy surfaces. The non-driven degrees of freedom for each point on the surface may be taken either from the most recently minimized structure or else from the starting structure.
- In force-field substructures, parameters may be given a geometric dependence on values of torsions, angles or ring size. This geometric dependence is retested at key points in a simulation, such as the start of a new step of conformational search.
- Ability to turn off interactions between sets of atoms. The sets are defined with the ASET command, and interactions are turned off with the ASNT command.
- VDWB command replaces bend interactions with van der Waals interactions, for modeling transition-metal complexes using the "points-on-a-sphere" model.
- In addition to harmonic constraints on atomic positions (FXAT), distances (FXDI), angles (FXBA), and dihedral angles (FXTA), we provide support for fully "frozen" atoms: atoms that do not move at all, but whose effect is felt by the atoms that can move. This is encoded by specifying a negative force-constant in a FXAT command.

Note: Starting with MacroModel 6.0, there were changes in the way simultaneous FXAT and SUBS commands are handled. See the SUBS command description on page 64.

- OPLS* and OPLS_2001 force fields now include short-range purely repulsive potentials on the polar hydrogen atoms. These potentials do not significantly affect the energies of normal structures but prevent catastrophic close approaches by negatively changed atoms in highly strained conformations such as those generated in aggressive conformational search methods such as MCMM, LMCS, and LMC2. This modification is not needed for the OPLS_2005 force field, which uses Lennard-Jones 12-6 potentials for interactions involving such atoms.

- Torsional potentials for highly strained conformations in which three of the atoms at one end of the torsion become collinear can cause problems during minimizations. The torsional potential is damped as either triplet of atoms at the end of the torsion becomes linear. The strength of the damping can be controlled via arg8 of `FFLD`.

1.2.2 Energy Minimization Methods

- Large and small molecules.
- Single or multiple structures.
- Full structure or substructure.
- Substructures can be specified using ASL expressions.
- Positional, distance, and angular constraints.
- Choice of steepest descent (SD), Powell-Reeves conjugate gradient (PRCG), truncated-Newton conjugate-gradient (TNCG), Oren-Spedicato variable metric (OSVM), full-matrix Newton-Raphson (FMNR), and Limited Broyden-Fletcher-Goldfarb-Shanno (LBFGS) minimization methods.
- Enhanced saddle-point searching (see the `MINI` command on page 52).
- Eigenvalue minimum testing and vibrational frequencies.
- Visualization of normal modes (`VIBR` and `VBR2`).
- Harmonic-entropy free-energy calculations.
- Greater control over intermediate output during minimization: see `MINI` command.
- Explicit saddle-point search method (see the `SDLP` command on page 61).
- The `FXAT` command has facilities which allow constraints to be relaxed in a stepwise manner. Minimization may be performed at each stage. This is useful in refining poorly initialized structures which may have been derived from NMR or low-resolution crystallography, or from homology modelling.
- The LCPO method, our accurate analytical surface-area computation method, is parameterized for metal atoms. `DEBG 91` uses the old surface area method.
- Additional LCPO parameters allow use of LCPO surface-area calculation with a broader variety of atom types.
- Automatic sequential minimization using multiple minimizer methods (`MWRT`).

1.2.3 Conformational Searching and Comparison

- Monte Carlo (MCMM) or systematic methods (SPMC).
- Low-mode search method (LMCS).
- Large-scale low-mode search method (LMC2). This facility extends the LMCS methodology to systems large enough to encompass entire proteins.
- ConfGen (CGEN). This is a separately-licensed module for rapidly and systematically generating conformations of ligand-like molecules.
- Acyclic, multicyclic, macrocyclic structures.
- Internal coordinate searching.
- Molecular docking (translation/rotation) searches.
- Local or global conformational searches.
- Duplicate conformer elimination.
- The MSYM opcode uses the numbering symmetry library mmsym which automatically and more generally identifies a suitable numbering order for use in comparing molecular conformations.
- Elimination of redundant conformers using positional and energetic comparisons (ADDC) without changing internal geometry. Jaguar energies may be used in this process.
- Automatic setup using the AUTO opcode is supported for Monte Carlo searches (MCMM), systematic searches (SPMC), low-mode searches (LMCS), large-scale low-mode searches (LMC2), and combinations of MCMM with LMCS or LMC2. Automatic setup may also be used to conduct separate searches on each structure in the input structure file (serial searches) for all of these methods except those involving LMC2. Serial searches employing LMC2 are not supported. The use of AUTO with LOOP is not supported, and if it is used with MBAE, it should be used with caution.
- Constrained searches.
- Ability to initialize a new search from the output of a previous, partial search.
- Ability to trigger a summary of search results so far while the program is running.
- LIGB command allows configurational search about a metal coordination center.
- LOOP. A fast method for generating candidate protein loop conformations based on the tweak method, using the sequence from the protein structure provided or an alternate sequence.

- Use of FLAP ring vertex reflection algorithm in MCMM and low-mode searches (FLAP).
- Use of ring templates for MCMM and low-mode searches (MCRC).

1.2.4 Molecular Dynamics and Monte Carlo Simulations

- Molecular Dynamics or Stochastic Dynamics.
- Multiple timestep acceleration.
- Constant temperature or energy.
- SHAKE bond length constraints.
- Translational and angular momentum constraints.
- Simulated annealing.
- Internal coordinate and surface area monitoring.
- Hydrogen-bond monitoring.
- Local conformational searching.
- Evaluation of average enthalpy.
- Mixed-mode Monte Carlo/Stochastic Dynamics procedure speeds exploration of conformational space.
- Velocity-Verlet integration scheme for greater efficiency of mixed-mode calculations.
- Importance sampling for Monte Carlo or mixed-mode calculations further speeds exploration of conformational space.

1.2.5 Coordinate Manipulation

- Different molecules may be aligned by center, principal axis, or both, using COPY and ALGN.

1.2.6 Molecular Association

- Embrace (MBAE) Multi-ligand Bimolecular Association with Energetics: A method for automatically minimizing or performing conformational searches on prepositioned ligands in the active site of a protein and obtaining energetic information related to the association of the ligands with the protein.

1.2.7 Partition Coefficients

- `LOGP`: A automated calculation method for estimating the logarithm of the partition coefficient for a series of solutes between two immiscible liquids.

1.2.8 Free-energy Perturbation

- See the `FEGA`, `FEIA`, `FEAV`, `FESA`, and `FESU` commands; also the didactic write-up in Chapter 15 of the *MacroModel User Manual*. Free-energy perturbation may be carried out using mixed-mode Monte Carlo / Stochastic Dynamics, as well as SD or MD. When used in network-distributed mode, each process runs a separate window.

1.2.9 Monte Carlo Simulation

- Internal coordinate Metropolis Monte Carlo.
- Structure sampling.
- Evaluation of average enthalpy.
- Mixed mode MD/MC simulations.
- Importance sampling for MC and MD/MC.
- A broad range of "smart" Monte Carlo and mixed MC/MD methods:

  Individual Methods:

  - SD: Stochastic Dynamics
  - ADF: All-Degree-of-Freedom Metropolis Monte Carlo
  - TMC: Torsional Monte Carlo
  - JBW: Jumping-Between-Wells importance sampling Monte Carlo

  Mixtures:

  - ADF/JBW
  - SD/TMC (original MCSD method)
  - SD/ADF
  - SD/JBW
  - SD/ADF/JBW

1.3 Computational Features

- Schrödinger products provide support for submitting jobs to batch queues. Any jobs that can be run on a remote machine can be submitted to a batch queue. See Chapter 6 of the *Installation Guide* for information on setting up batch queue support.

- MacroModel jobs can be run with a simplified input file that contains keyword-value pairs. These jobs can be submitted with the `$SCHRODINGER/macromodel` command.

- Reads/writes old and new file formats. With Maestro, we introduce a new m2io file format that is self-defining and extensible. All programs are back-compatible with our old mmio file format for both reading and writing; however, the new format has the ability to store additional data, such as graphical representation of atoms between Maestro sessions (CPK, wireframe, etc.). Indeed, the `m2io` format can be used to store arbitrary data associated with molecules, atoms or bonds. Starting in MacroModel 8.0, Maestro format files are written by default regardless of the format of the input structure file. However, you can still direct MacroModel to write old MacroModel format files by specifying `DEBG 6`.
- The `TIME` command reports CPU time(user+system) since the previous invocation of `TIME`, or (on the first invocation) since program start.
- The `DEBG 1000` output of task timings is more detailed and informative.
- Vectorized first derivative, second derivative, and energy calculating modules.
- Network-distributed parallel processing for individual searches (`MCMM`, `LMCS`, `LMC2`, and mixed `MCMM/LMCS` and `MCMM/LMC2`).
- Network-distributed parallel processing for serial calculations in which many input structures are sequentially processed in similar ways. Two mechanisms are supported for performing these calculations. One functions via commands such as `NPRC` placed in the `.com` file; the other is a perl-based wrapper script for MacroModel called `para_bmin`. Both mechanisms support distributed processing of serial searches (`MCMM`, `LMCS`, mixed `MCMM/LMCS`, `SPMC`, `CGEN`) and the former also supports multiple minimizations and `MBAE` calculations (minimizations and searches).
- Compressed file format is supported for input and output Maestro files. Compression of multiconformer files greatly diminishes disk requirements.
- `RWND` command allows the input or output file to be rewound and used as the input to another procedure; only minimization is supported right now for the second procedure. We expect this to be most useful for reminimizing the output of a conformational search.
- During a repetitive procedure, you can specify behavior if an iteration fails (`BGIN`).
- The format of the `.com` file is checked when it is read in and warning messages are produced in the `.log` file for statements that are likely to be misinterpreted by MacroModel due to formatting problems. If `DEBG 960` is specified this reporting of format problems is suppressed. If `DEBG 961` is specified then the job terminates immediately when such a warning is produced.
- When solvation or force field parameters are missing for a structure in a serial calculation the program continues with the next structure to be processed rather than stopping, unless `DEBG 49` is specified.

1.4 Program Capacity

MacroModel uses dynamic memory allocation. For most computations, the size of the system that can be studied is limited only by the resources of the computer on which the simulation is performed. Some compiled-in limits remain, and a few are described below:

- MAXFV—The maximum number of atoms that can be used in certain second-derivative procedures. Opcodes that are affected by this limit are documented in the opcode descriptions. For this version of MacroModel MAXFV is 1100.
- The maximum number of comparison atoms that can be specified is 20,000. In addition, the total number of fixed and frozen atoms cannot exceed 20,000. Structural restraints can be applied to sets of atoms, but the limit is 20,000 each of stretches, bends, and torsions.

The maximum number of structures that can be saved in a conformational search or multiple minimization is under user control. See:

- Arg1 of MCMM, LMCS, LMC2, and SPMC
- Arg2 of MULT
- Arg5 of LOOP

1.5 Miscellaneous

- The SPAT command allows the user to instruct MacroModel how to proceed when it encounters certain atom types.
- The GEOM command allows the user to obtain geometric information about the molecule from the MacroModel command file.
- Many properties are now recorded in Maestro-formatted output structure files. These properties are displayed in Maestro's project table.

1.6 Citing MacroModel in Publications

The use of this product should be acknowledged in publications as:

MacroModel, version 9.7, Schrödinger, LLC, New York, NY, 2009.

Running MacroModel

2.1 Running MacroModel From the Command Line

MacroModel jobs may be started from the Maestro GUI or from the command-line. The mechanisms for running jobs from the command line are described in this chapter. You might wish to run MacroModel from the command line for any of the following reasons:

- To exercise greater control over MacroModel behavior: not all of MacroModel's functionality is available from Maestro.
- To perform a series of related MacroModel runs: it may be easier to edit an existing MacroModel command file than to set up each job separately from Maestro. Sometimes several jobs are queued up and run from a shell script.
- To debug an aberrant run: when a MacroModel run fails to do what is expected, a simple command can sometimes be added to the MacroModel command file to instruct the program to give more detailed output.

MacroModel jobs run under Schrödinger's Job Control facility. This facility manages the execution and monitoring of jobs, and handles the input and output files and the incorporation of results into a Maestro project. The *Job Control Guide* describes how to set up the information needed for Job Control to run on the computers to which you have access. It includes information on remote hosts, clusters, and batch queues.

2.1.1 Environment Variables

Whenever MacroModel is run, the UNIX environment variable `SCHRODINGER` must be set to the directory where MacroModel was installed. In addition, there are other environment variables that can be set to override default resource values. See Appendix B of the *Job Control Guide* for more information.

By default, the Schrödinger job control facility uses `ssh` to communicate between remote nodes. A mechanism exists to specify alternative commands for remote command execution. The environment variable `SCHRODINGER_RSH` should be set to the alternate application, which need to support the same basic command-line syntax as `ssh`. Commands need to be passed without asking for a password or requiring any other user interaction. For more information, see Section 6.2 of the *Installation Guide*.

2.1.2 Command-line Syntax

MacroModel is run from the command-line by executing the shell script `bmin`, which is located in the `$SCHRODINGER` directory, or the `macromodel` script, also located in this directory. The `macromodel` script and its input file are described in Chapter 3; this section focuses on the `bmin` script. The `bmin` script determines the proper executables to run for the host being used. Additional utility programs are provided in the `$SCHRODINGER/utilities` directory. The `$SCHRODINGER/macromodel-v`*version*`/data` directory also contains a number of files (such as force field files) that MacroModel needs in order to run.

Whether MacroModel is started from the shell or from Maestro, if copies of the force field, solvent or atom type files exist in the `$HOME/.schrodinger` directory, they override the versions of these files in the `$SCHRODINGER/macromodel-v`*version*`/data` directory. If copies of the force-field, solvent and/or atom-type files exist in the local directory, they override any versions of these files in both the `$HOME/.schrodinger` and `$SCHRODINGER/macromodel-v`*version*`/data` directories.

The syntax for the `bmin` command is as follows:

`$SCHRODINGER/bmin [`*options*`]` *jobname*

The options for the `bmin` command are listed in Table 2.1. The standard Job Control options are also supported—see Section 2.3 of the *Job Control Guide*. Additional diagnostic options, which do not run the job but provide information, are also given in the section mentioned. In particular, you should note the syntax of the `-HOST` option, which is used to specify the list of hosts used for the job.

Table 2.1. Options for the bmin command.

Option	Description
`-DEBUGGER` *debugger_name*	Run `bmin` in the foreground with a debugger called *debugger_name*.
`-DEBUG2`	Print out debugging information from the scripts used to start `bmin`.
`-h[elp]`, `-HELP`	Show usage message.
`-HOSTFILE` *hostfilename*	The name of the hosts file to use for this run. Default is `schrodinger.hosts`.
`-INTERVAL` *value*	The maximum time in seconds for updating the monitoring files.
`-LOCAL`	Do not use a temporary directory keep the files. Keep files in the local directory.
`-NO_REDIRECT`	Run in the foreground and send output to standard output.

Table 2.1. Options for the bmin command. (Continued)

Option	Description
-TEST	Run the standard test suite (jobname not needed).
-DEV_TEST	Run the developers version of the standard test suite (jobname not needed). Turns on -TEST.
-WAIT	Do not display a prompt until the job finishes.

Note: The -NO_REDIRECT command-line option to the $SCHRODINGER/bmin script is incompatible with distributed jobs.

Jobs are automatically run in the background, and the UNIX command prompt is displayed without waiting for the job to finish. The -WAIT option can be used to prevent the command prompt from being displayed until after the job finishes:

```
$SCHRODINGER/bmin -WAIT jobname
```

The actual MacroModel process is still run in the background, so pressing CTRL+C or CTRL+Z does not affect it.

2.1.3 Storage of Temporary Files

By default, MacroModel jobs are run from a scratch directory, rather than from the one in which the job was started up. The scratch directory is generally on the local file system. A detailed description of how Job Control selects the scratch directory is given in Section 2.4 of the *Job Control Guide*.

When the job is started, its input files are copied to the scratch directory. While the job runs, temporary job files are maintained in the scratch directory. During the run, the .log file is periodically copied back to the directory from which the job was started. When job monitoring from Maestro is in effect, structural monitoring files are also copied back periodically. These show how the structure is changing during the job. You can specify the interval in seconds at which these files are copied back with the -INTERVAL option. When the job finishes, the output files are copied back to the original startup directory.

MacroModel runs from the job startup directory if you use the -LOCAL option when running the job:

```
$SCHRODINGER/bmin -LOCAL jobname
```

See Section 2.2 on page 17 for a description of the how to use the job control facility to interact with running MacroModel jobs.

2.1.4 The .com File

To run a MacroModel job, a command file (`.com` file) and a molecular structure file are required. The `.com` file generally has the name *filename*`.com`, where *filename* is a valid filename prefix. The `.com` file contains the name of the input structure file, the intended name of the output structure file, and a list of operation codes ("opcodes") telling MacroModel which operations are to be performed and in which order they are to be performed.

The general format of a command file for MacroModel is:

```
filename.mae
filename-out.mae
 OPCD  IIIIII IIIIII IIIIII IIIIII FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF
 OPCD  IIIIII IIIIII IIIIII IIIIII FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF
 OPCD  IIIIII IIIIII IIIIII IIIIII FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF
```

The example above has been modified below to show the column numbers:

```
filename.mae
filename-out.mae
12345678   13     20     27     34     41    47   52    58   63    69   74
||||||||    |      |      |      |      |     |    |     |    |     |    |
VVVVVVVV    V      V      V      V      V     V    V     V    V     V    V
 OPCD  IIIIII IIIIII IIIIII IIIIII FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF
 OPCD  IIIIII IIIIII IIIIII IIIIII FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF
 OPCD  IIIIII IIIIII IIIIII IIIIII FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF
```

For comparison, column numbers have also been added to the example below, which is taken from an actual `.com` file:

```
filename.mae
filename-out.mae
12345678   13     20     27     34     41    47   52    58   63    69   74
||||||||    |      |      |      |      |     |    |     |    |     |    |
VVVVVVVV    V      V      V      V      V     V    V     V    V     V    V
 MMOD       0      1      0      0     0.0000     0.0000     0.0000     0.0000
 FFLD      10      1      0      1     1.0000     0.0000     0.0000     0.0000
 BDCO       0      0      0      0    20.0000 99999.0000     0.0000     0.0000
 READ       0      0      0      0     0.0000     0.0000     0.0000     0.0000
 CONV       2      0      0      0     0.0500     0.0000     0.0000     0.0000
 MINI       1      0    500      0     0.0000     0.0000     0.0000     0.0000
```

This file would have a name such as *jobname*`.com` and, if so, would be submitted from the command prompt with a command such as

`$SCHRODINGER/bmin` *job_name*

Maestro uses a similar command for computations prepared with the interface.

2.1.4.1 File Specifications

The first line of the instruction file above is the name of the input structure file. The resulting output structure file is given the name listed on the second line of the instruction file. The input

structure file contains the molecular structure(s) of the molecules to be processed, and the output file will contain the structures after processing. The input and output structure files can be named with any valid UNIX filename, as long as they differ.

Note: The output structure file now uses the Maestro format regardless of the format used in the input structure file. Specifying `DEBG 6` forces MacroModel to use the MacroModel format for the output structure file.

In previous versions of MacroModel, the input and output structure file names were usually named *filename*`.dat` and *filename*`.out`. Now that all Maestro formatted structure files are, by default, named with the `.mae` extension, Maestro now automatically names the input structure file *filename*`.mae` and the output structure file *filename*`-out.mae`. We recommend using *filename*`.mae` and *filename*`-out.mae` for these files when Maestro-formatted files are used, so that the suffix reflects the format used in the file. The *filename* stem for the input file is often the same as the *filename* prefix for the `.com` file. You can also use compressed Maestro files (`.maegz` or `.mae.gz`) for both input and output.

In addition, any input substructure (`.sbc`) or velocity (`.vel`) files should contain the same stem as the input structure file. Similarly the output energy listing (`.mmo`), substructure, and dihedral drive (`.grd`) files should have the same stem as the output structure file. The output `.log` file contains the same stem as the `.com` file. Experienced users of MacroModel should be aware that this mechanism has not changed from previous versions of the program.

The program allows full-path filenames up to 100 characters in length. The *filename* stems need not be identical for the input, output, and `.com` files, but they are the same in jobs set up in Maestro and it is usually most convenient to follow this convention. MacroModel may create various temporary files in the course of its work. The *filename* stem for these files is taken from the *filename* stem of the input file name.

2.1.4.2 Operation Code Specifications

The remaining lines in the `.com` file provide the instructions to MacroModel about the type and order of calculations to be performed. The opcode lines must be of the following fixed format:

```
 OPCD  IIIII IIIII IIIII IIIII FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF FFFFF.FFFF
```

`OPCD` is a four-letter operation code ("opcode") for a command; *each line containing an opcode begins with a space*. The eight fields after the `OPCD` are referred to as "arguments." `IIIIII` represents an integer argument in Fortran I6 format (arg1-arg4), and `FFFFF.FFFF` represents a floating point argument in F10.4 format (arg5-arg8).

Note: Arg5-arg8 require a decimal even when integer values are provided. Each opcode defines a general command for MacroModel, and the integer and real-number arguments specify more specific operations or options. Any line containing an opcode can

be turned into a nonexecutable comment line by placing a character other than a blank space in the first column. This is useful for "commenting out" existing commands as well as for adding explicit comments. Note that tabs are *not allowed* in `.com` files.

Note: The form of the `.com` file must correspond *exactly* to that shown above. All integer and real arguments must be aligned as shown. We recommend using an existing command file as a template, rather than building one from scratch. Many of the arguments have default values which make it unnecessary to explicitly include them in the `.com` file if the defaults are the desired values. Omitting an argument causes the default value to be used. Using a value of zero for an argument generally is also interpreted as a request for the default value. The actual values for the default parameters are indicated in the descriptions of the opcodes.

Though there are exceptions, opcodes are generally processed in the order in which they are given in the `.com` file; thus, the order of the command lines is important. Notes about command order are given as part of the opcode descriptions. Examples of valid `.com` files for numerous common modeling tasks are given in the *MacroModel User Manual*. Though few users learn how even the most simple opcodes work at first, it becomes more necessary to do so as the complexity of the modeling task increases.

2.1.4.3 Using an Executable .com File

If you have a command file that must be run repeatedly (such as the MacroModel test scripts), you can embed the contents of the command file in an executable shell script. Below is an example:

```
#! /bin/csh -f
cat <<EOC > a_run.com
jobname.mae
jobname-out.mae
 FFLD
 READ
 ELST       2
EOC
$SCHRODINGER/bmin a_run
```

If the above contents were contained in an executable file called `mm2-ELST`, MacroModel could be invoked to run the included commands by simply naming the file on the command line, e.g.,

```
mm2-ELST &
```

2.2 Job Control

Schrödinger's job control facility controls the execution of MacroModel jobs, including remote and distributed MacroModel jobs (see Section 2.3 on page 19). Whether a job is started from the shell or from Maestro, the job control procedures can be carried out either from the shell, as described below, or from the Monitor panel in Maestro. For a complete description of job control, see Chapter 3 of the *Job Control Guide*.

The `$SCHRODINGER/jobcontrol` script provides a variety of mechanisms for obtaining information about jobs and for interacting with running jobs. The script can be run either interactively or as a shell command. Executing the command:

```
$SCHRODINGER/jobcontrol
```

without arguments invokes the interactive use of the facility; the `-HELP` option describes usage. The `-stop` *jobid* option stops a job gracefully. The `-pause` and `-resume` options suspend and restart jobs, and the `-update` option causes the running job to perform an update to its log file.

These functions supersede the former behavior of the `.stp`, `.hlt`, and `.upt` files (described in Section 2.2.1 on page 17) which no longer have an effect unless `DEBG 931` is specified.

Note: While the `-kill` option can be used to stop a job immediately, its use with distributed MacroModel jobs might on occasion not kill all of the processes associated with the run. We recommend using the `-stop` option to avoid this problem.

The job control utility maintains a database containing the status of all jobs in `$HOME/.schrodinger/.jobdb`. Normally, when running MacroModel jobs from Maestro's project facility, the database is cleared when jobs are incorporated into projects. Otherwise, the `.jobdb` directory could become quite full over time. The `jobcontrol` utility can be used to purge the job database. For example, to purge the entire database, use the `-purge all` option. Of course, you should not purge jobs that you expect to incorporate into Maestro projects. Completed jobs are purged by default after a week. The job control utility checks for old jobs whenever a job is initiated. To change the threshold for purging jobs, set the environment variable `SCHRODINGER_JOBDB_CLEANUP` to the desired value in seconds.

2.2.1 Interacting With a Running MacroModel Job Via Files

The job control facility can be run either from Maestro or from the command line, and can be used to interact with running MacroModel jobs. Using the job-control facility is the recommended method for interacting with running MacroModel jobs, but file-based methods for controlling running MacroModel jobs are described below.

The facilities described in this section become inactive if `DEBG 930` is specified, which is the default behavior. `DEBG 931` may be specified to enable these facilities.

You can interact with a running MacroModel job by creating a file in the directory from which the job was launched. One simple way to do this from the UNIX command line is to use the UNIX `touch` command:

`touch` *filename.sfx*

where *filename* is the filename prefix of the input file (usually the job name) and *.sfx* is a suffix that depends on the action desired, as described in the following sections.

2.2.1.1 Stopping MacroModel

You may wish to stop a MacroModel run before it has completed normally. For example, scrutiny of the log file generated so far may reveal that a conformational search is essentially complete before the requested number of conformations have been generated. The job may of course be killed from the UNIX shell by using the command

`kill -9` *pid*

where *pid* stands for the process identification number, which can be obtained through use of the UNIX `ps` command. If this is done, then MacroModel exits without doing its normal cleanup. This means, among other things, that output buffers will not be flushed, and that scratch files, such as *filename*`.tmp`, will not be converted to final form, such as the ASCII file *filename*`-out.mae`, and removed. This state of affairs also occurs when the system kills the MacroModel job or when the system crashes. Recovery from this state is possible by means of the `TRED` command, described in Chapter 4.

There is, however, a gentler way to kill MacroModel—one which causes the process to carry out its normal clean-up operations. To do this, create a file called *filename*`.stp` in the directory in which the MacroModel job was initiated. The `.stp` suffix is a mnemonic for "stop." MacroModel periodically checks for the existence of such as file, and, if it finds it, branches immediately to its normal cleanup and exit procedure. MacroModel removes the `.stp` file prior to exiting; thus the same job may be rerun if desired without first removing this file manually.

2.2.1.2 Putting MacroModel to Sleep

Occasionally you might like to temporarily suspend the execution of a MacroModel job and restart it later. One reason for doing this might be to obtain faster response time for an interactive program, such as Maestro. This can be accomplished by creating a file in the directory from which MacroModel was started called *filename*`.slp`; the `.slp` suffix is a mnemonic for "sleep," and *filename* is the prefix of the input file. MacroModel periodically looks for the `.slp` file, and, if it finds it, suspends execution, except for periodically attempting to verify the

file's continued existence. When it fails to find the `.slp` file, it resumes execution. Thus, to resume execution of the MacroModel job, remove the `.slp` file using the UNIX `rm` command.

The UNIX `kill` command can also be used to stop a MacroModel process; to stop the process, type:

```
kill -STOP pid
```

To restart it later, type:

```
kill -CONT pid
```

where *pid* is the process identification number of the job.

When a job is put to sleep the CPU suspends execution and the job may in time be swapped out of active memory (RAM); however, it continues to occupy swap space on disk. If, in a particular situation, the availability of swap space is limiting the number of jobs that can be simultaneously run, putting one or more of them to sleep will not help. This option is, however, useful for obtaining more responsive performance when CPU availability, rather than swap space, is limiting. There are less drastic means of lowering the priority of a job as well—namely, `renice` and `npri`—but these require special privileges if it is desired to restore the original priority.

2.2.1.3 Triggering an Interim Job Summary

If the program detects a file whose name is *filename*`.upt`, it will, at the next convenient point, print to the log file a summary of its progress so far, then remove the `.upt` file. In the current version of MacroModel, this facility is implemented only for conformational search. If Maestro is running, the update information also appears in the Monitor panel. When MacroModel is run as a standalone process, the user can trigger an update manually by simply creating the `.upt` file.

2.3 Network-Distributed Processing

Some types of MacroModel calculations may be split into largely independent tasks which are run separately on different processors or computers across a network. The number of tasks may be greater than or equal to the number of processors requested, so that processors carry out more than one task before the job finishes. Distributing the overall calculations across multiple processors (a form of parallel processing) can significantly reduce the time required to complete the job.

2.3.1 Load Balancing

Load balancing involves giving different processors different amounts of work depending on how rapidly they carry out tasks assigned to them. Often if the overall calculation requires a lot of CPU time, there is a distinct advantage to setting the number of tasks to a small multiple (e.g. greater than 3) of the number of processors requested, particularly when the processors differ in inherent speed and load (number of programs contending for CPU time). When a processor finishes a task, it is assigned a new one from the pool of unprocessed tasks until the overall calculation is complete. As a result, processors that return results faster are assigned a greater portion of the overall calculation.

2.3.2 Types of Distributed Processing

MacroModel supports two general classes of distributed processing. In the first, a calculation on a single molecular system is split up into a number of tasks. FEP calculations, minimization of conformers, and conformational searching methods supported by MacroModel (including `MCMM`, `LMCS`, `LMC2`, mixed `MCMM-LMCS`, and mixed `MCMM-LMC2`), can be carried out using this paradigm. For searches, intermediate results from the first tasks can be incorporated and used when additional tasks are initiated. The second class of calculation, referred to as a serial calculation, involves preforming independent yet similar calculations on many different systems (e.g. a separate conformational search on each of 1000 ligands in the input structure file). Due to the independent nature of the calculations each task carries out the calculation for a subset of the systems provided. The tasks that can be distributed using MacroModel include: serial `MCMM`, `LMCS`, `LMC2`, `MCMM-LMCS` `MCMM-LMC2`, `SPMC`, and `CGEN` conformational searches, as well as multiple minimizations of non-conformers (i.e. without checks for redundant conformers). In addition, both minimization and conformational search `MBAE` calculations for multiple ligands can be distributed due to their serial-like character.

2.3.3 MacroModel-Based Distributed Processing

With the exception of MacroModel/ConfGen searches initiated from the Phase panels, Maestro does not support distributed processing using MacroModel. Under most circumstances distributed MacroModel jobs must be run from the command line. If any of the processors used for a distributed MacroModel job are on different computers than the one on which the command is initiated, it is a remote job. It is often necessary to prepare these computers, your accounts on them, and some files (e.g. a `schrodinger.hosts` file). See Chapter 6 of the *Installation Guide* to for information on how to carry out such preparations.

MacroModel has two mechanisms for carrying out network-distributed processing. In the first, the setup, submission, monitoring, and collection of results is carried out internally within MacroModel's back-end program, `bmin`. In the second, these steps are carried out externally

using the para_bmin command, which in turn uses bmin to process individual tasks. The internal approach is more general, but takes more effort to set up.

For either method, a starting point is the set of files needed to carry out the calculation without distributing it. For many classes of calculations, these can be conveniently constructed using MacroModel panels in Maestro and clicking the Write button to save the files needed for the calculation. While a distributed MacroModel job is running, it creates a number of temporary files in the directory where intermediate results are stored. These files may appear in your current directory. These files are removed automatically at the end of a successful run but are needed while the run is underway. Removing or altering them during the run may affect the results or cause the run to fail.

2.3.3.1 Internal Distribution

Internal distribution is carried out by MacroModel's back-end program, bmin. All types of distributed calculations supported by MacroModel, with the exception of CGEN searches, may be carried out using this approach:

- MULT—Multi-conformer minimizations.

  Note: Currently, distributed multi-conformer minimizations require inclusion of the MULT opcode in the command file.

- MCMM conformational searches including single structure searches, multi-conformer seeded searches, and serial MCMM searches.
- LMCS—Low-mode conformational search computations, including single-structure searches, multi-conformer seeded searches, and serial LMCS searches.
- LMC2—Large-scale low-mode calculations based on a single input seed structure.
- MCMM/LMCS and MCMM/LMC2 mixed low-mode conformation searches, including single-structure searches, multi-conformer seeded searches, and serial LMCS searches.
- Serial MCMM, LMCS, and mixed-LMCS/MCMM conformational searches.
- MBAE—Protein-ligand minimizations and conformational searches.
- FEAV, FESA—Free-energy perturbation calculations.

For internally distributed calculations, the input files for the corresponding non-distributed calculation must be modified by including an additional line containing the NPRC operation code. The first argument to NPRC gives the number of processors requested, the second argument indicates the amount of work to assign to each task, and the third indicates the cycle time for monitoring in seconds. 60 is a reasonable choice for argument three, but longer times may be appropriate, particularly if the network is slow. If the fourth argument is nonzero, an initial

test calculation is performed on each of the hosts (this is recommended). Simple examples for each of the supported types of distributed calculation are located in the directory: `$SCHRODINGER/macromodel-v`*version*`/test/dbmin` after installation.

After the `NPRC` line has been added to the `.com` file, MacroModel is run in the normal manner, e.g.:

`$SCHRODINGER/bmin` *test1*

Note: The `-NO_REDIRECT` command-line option to the `$SCHRODINGER/bmin` script is incompatible with distributed jobs.

If a distributed MacroModel job is to be submitted to a queue, the following syntax should be used:

`$SCHRODINGER/bmin -LOCAL -HOST` *queue_name jobname*

The `-LOCAL` option keeps intermediate files in the local directory and facilitates inter-node communication.

Note: For the `-LOCAL` option to function properly, it is often necessary for the current directory to be mounted on all computers involved in the process of running the job, including the machine on which the job is run and the machine that queues the job.

The *queue_name* argument to the `-HOST` option specifies the name of the queue on which the job should be run. Using the example `schrodinger.hosts` file in Section 2.3 on page 19, *queue_name* could be replaced with `queue2`.

The *jobname* is the name of the command file for the job, without the `.com` extension.

Note: It is currently necessary to submit such jobs from an NSF-mounted directory.

A complete description of the syntax for running MacroModel from the command line can be found in Section 2.1.2 on page 12.

Up to a total of 100 processors on up to 100 remote hosts can be requested for a single calculation. MacroModel looks through the hosts file for up to the number of processors requested by the `NPRC` opcode, starting from the first entry. If insufficient hosts are listed, a message appears in the `.log` file and the run continues, using the hosts listed. If the test calculation is requested via arg4, it is performed first. If there are significant differences amongst the energies returned by the initial test calculations, the job stops and provides information as to which hosts gave different answers. Since internally-distributed MacroModel calculations select hosts starting from the top of the `schrodinger.hosts` file, it can be advantageous to use customized files for different calculation. See Section 6.1 of the *Installation Guide* for information on setting up a `schrodinger.hosts` file.

2.3.3.2 External Distribution

External distribution of MacroModel jobs supports serial calculations including serial MCMM, LMCS, LMC2, MCMM-LMCS MCMM-LMC2, SPMC, MBAE, and CGEN conformational searches. Since processing each input structure in these calculations is inherently independent of the other structures, the collection of input structures can be broken down in to smaller sets and processed by separate MacroModel runs using bmin. This could be done by the user manually or by a custom script. The para_bmin utility is provided to carry out this task. The para_bmin jobs use the same input files as the corresponding non-distributed calculation and do not require that customized schrodinger.hosts files be made up when jobs are run on different collections of processors or hosts.

To launch a distributed job, enter the following command in a terminal window:

$SCHRODINGER/para_bmin [*options*] *jobname*

replacing *jobname* with the stem of the input file name (not needed if the -TEST or -HELP options are specified) and including any of the options listed in Table 2.2. The standard Job Control options are also supported—see Section 2.3 of the *Job Control Guide*. Additional diagnostic options, which do not run the job but provide information, are also given in the section mentioned.

Table 2.2. Options for the para_bmin *command.*

Option	Description
-HELP|-help|-h	Print the para_bmin command syntax, options, and their definitions, and exit.
-HOSTFILE *hostfilename*	The name of the hosts file to use for this run. The default is the schrodinger.hosts file found by Job Control.
-INTERVAL *time*	The maximum time in seconds for updating the monitoring files.
-LOCAL	Do not use a temporary directory keep the files. Keep files in the local directory.
-GZIP	Compress output structure files.
-JOBCTS *maxctsjob*	Ensure that each subjob has no more than this many structures to process. Default: 100.
-NJOBS *number*	Divide the overall job into this many subjobs.
-OUTPUT_ORG *option*	Produce more than one output structure file. If option is BY_SUBJOB produce one output file for each subjob. Otherwise create a new directory called *option* and create separate output files for each input structure within this new directory. Automatically sets -LOCAL if a new directory is created.

Table 2.2. Options for the `para_bmin` *command. (Continued)*

Option	Description
`-ROBUST` *number*	Robust execution of `bmin`: `0` off `1` on Default: `1`
`-SUBINTERVAL` *time*	The maximum time in seconds for updating the monitoring files of subjobs. (Default: do not monitor subjobs)
`-WAIT`	Wait until the job finishes before executing the next command.

In particular, you should note the syntax of the `-HOST` option, which is used to specify the list of hosts used for the job.

It is common to specify both the number of tasks (`-NJOBS`) and the number of processors to use. To request that a calculation be divided into 30 tasks and use 4 processors on `computer1`, 1 on `computer2`, and 5 on `computer3`, you would enter the command:

```
$SCHRODINGER/utilities/para_bmin -HOST "computer1:4 computer2:1 computer3:5"
-NJOBS 30 jobname
```

If the number of processors for a particular computer is omitted, a default of 1 is used. The number of tasks is reset to the number of input structures, if there are fewer structures in the input structure file than tasks. If the number of tasks requested results in more than the `maxctsjob` (default 100) structures in each task, the number of tasks is increased to (number of input structures) / `maxctsjob`. The computers listed in the `host_list` must be described in the `schrodinger.hosts` file used for the calculation. The process that controls the overall job is run on the first host listed in `host_list`.

If the `-OUTPUT_ORG` option is omitted, all output structures are placed in the output structure file listed in the `.com` file, typically *jobname*`-out.mae`. However, our software will fail if an attempt is made to write an output structure file larger than 2 GBytes. The `-OUTPUT_ORG` option instructs `para_bmin` to create multiple output structure files and provides a mechanism to avoid this file size limit problem. If `-OUTPUT_ORG BY_SUBJOB` is specified, each task started up by `para_bmin` results in a separate output structure file:

jobname`-out_subjob_#.mae`

where # is the number of the subjob.

Alternatively, specifying `-OUTPUT_ORG` *directory* results in a separate output structure file for each input structure in the directory *directory* (as long as *directory* is not "`BY_SUBJOB`") which is created by the `para_bmin` run if it does not already exist. These files are typically called *jobname*`-out_ct_#.mae` where # is a count of the output structure files produced. This choice

also sets `-LOCAL`, which means that files are written locally, and therefore a remote `para_bmin` job must have the launch directory mounted on the remote host.

Running MacroModel with the Simplified Input File

MacroModel command input has come from a file that is structured in a fixed format, with integer and real numbers as the fields in each line of the file. This format is described in detail in . To provide a more flexible and readable interface, a new, simplified input file format has been designed. It is hoped that this format will make it easier for people to learn and work with the files that control MacroModel calculations. Along with this new input file format, a new command, `macromodel`, has been introduced that reads this input file. This chapter describes the new command and input file format. This is a limited implementation, so not all options or types of MacroModel calculations are supported.

The syntax of the `macromodel` command is as follows:

`$SCHRODINGER/macromodel [`*options*`]` *inputfile*

where *inputfile* can be a `.com` file or a simplified MacroModel input file. There is no required extension for the simplified input file. The options are described in Table 3.1. In addition to these options, the standard Job Control options listed in Table 2.2 of the *Job Control Guide* are supported, as are the `-LOCAL` and `-WAIT` options described in Table 2.3 of the *Job Control Guide*.

Table 3.1. Options for the macromodel command.

Option	Description
`-v`	Display the version number and exit.
`-h[elp]`	Print help message and exit.
`-NJOBS` *n*	Divide the overall job into *n* subjobs. A value greater than 1 forces a driver job to be submitted. Default: 1.
`-JOBCTS` *m*	Ensure that each subjob has no more than *m* structures to process. Default: 100.
`-NOJOBID`	Do not run the job under Job Control. With `NJOBS` > 1, this option only applies to the driver job.

The keywords for the simplified MacroModel input file are given in Table 3.2.

Table 3.2. Keywords for the simplified MacroModel input file

Keyword Syntax	Description
`INPUT_STRUCTURE_FILE` *filename*	Name of the input structure file. NO DEFAULT.
`FORCE_FIELD {MM2*|MM3*|AMBER*| AMBER94|OPLS|MMFF|MMFFs| OPLS_2001|OPLS_2005}`	Force field. Default: `OPLS_2005` . See the `FFLD` opcode.
`SOLVENT {None|Water|Octanol| CHCL3}`	Solvent to use (continuum solvation model). Default: `Water`. See the `SOLV` opcode.
`DIELECTRIC_CONSTANT` *const*	Dielectric constant. Default: 1.0. See the `FFLD` opcode.
`ELECTROSTATIC_TREATMENT {Constant dielectric| Distance-dependent| Force-field defined}`	Electrostatic treatment. Default: `Constant dielectric`. See the `FFLD` opcode.
`CHARGES_FROM {Force field| Structure file}`	Source of partial charges to use for electrostatics. Default: `Force field`. See the `CHGF` opcode.
`CUTOFF {Normal|Extended|None}`	Default: `Normal`. See the `EXNB` opcode.
`MINI_METHOD {PRCG|TNCG|OSVM| SD|FMNR|LBFGS}`	Minimization method. Default: `PRCG`. See the `MINI` opcode.
`MAXIMUM_ITERATION` *n*	Maximum number of iterations in a minimization. Default: 5000. See the `MINI` opcode.
`CONVERGE_ON {Gradient|Energy| Movement|Nothing}`	Quantity to use for testing the convergence of the minimization. Default: `Gradient`. See the `CONV` opcode.
`CONVERGENCE_THRESHOLD` *thresh*	Threshold for minimization convergence. Default: 0.05. See the `CONV` opcode.
`CONFSEARCH_METHOD {MCMM| Confgen|Lowmode|Mixed|mcmm| cgen|lmod|lmcs}`	Conformational search method. Default: `MCMM`. See the `MCMM`, `LMCS`, `LMC2`, and `CGEN` opcodes.
`MULTI_LIGAND {True|False}`	Specify whether the input contains multiple ligands (rather than a set of conformers). Used for serial searches. Default: `True`.
`INCONFS_PER_SEARCH` *nconf*	Number of input conformations per search. Default: 0. See the `MCOP` opcode.

Table 3.2. Keywords for the simplified MacroModel input file

Keyword Syntax	Description
`OUTCONFS_PER_SEARCH` *mconf*	Number of output conformations per search. This value must be a positive integer. Default: 1. See the `MCOP` opcode.
`CONFSEARCH_STEPS` *n*	Number of steps to use in the conformational search. Default: 200. See Section 4.10 on page 83.
`CONFSEARCH_STEPS_PER_ROTATABLE` *n*	Number of steps to use per rotatable bond in a conformational search. Default: 50. See the `AUOP` opcode.
`ENERGY_WINDOW` *value*	Energy window for keeping conformers. Default: 5.0. See the `DEMX` opcode.

Below is an example simplified file for a conformational search:

```
INPUT_STRUCTURE_FILE structure_path.mae
CONFSEARCH_METHOD mcmm
FORCE_FIELD OPLS_2001
SOLVENT Water
DIELECTRIC_CONSTANT 1.0
CHARGES_FROM Force field
CUTOFF None
MINI_METHOD PRCG
OUTCONFS_PER_SEARCH 1
CONFSEARCH_STEPS 200
MAXIMUM_ITERATION 500
CONVERGE_ON Gradient
CONVERGENCE_THRESHOLD 0.05
ENERGY_WINDOW 5.0
CONFSEARCH_STEPS_PER_ROTATABLE 50
ELECTROSTATIC_TREATMENT Constant dielectric
```

Operation Codes

This chapter provides reference information for using the operation codes. Unless otherwise specified, a zero in any opcode argument specifies the default value, if there is a default. Also, unless otherwise specified, the default is a reasonable value to choose if you do not have a good reason to select a different value.

4.1 File Reading and Writing

READ — READ a structure

Read a structure from the input file. Repeated READ commands read multiple structures from the input file.

arg1 *Molecular information retention*

0 Default behavior

–1 Clear existing molecular information of previously read molecules and re-evaluate the substructure information.

This value is useful while using READ in BGIN–END loops and using ASL syntax in .sbc files (see ASL1, ASL2 and ASL3).

–2 Clear existing molecular information of previously read molecules *without* re-evaluating the substructure information.

WRIT — WRITe a structure

Writes the current structure to the output file. WRIT is not necessary for Monte Carlo searches and energy minimizations. It is used with molecular dynamics. Multiple WRIT statements put multiple structures into the output file.

TRED — Temporary file REaD

Reads a set of atomic coordinates from *inname*.tmp. This function is used to recover structures from a failed MULT or MCMM run or when MULT is set in a multiple-loop MacroModel command file. In the latter case, all loops after the first must use a TRED rather than a READ. With Version 4.5 and later, the need for this command should be minimal, since updates are now performed periodically during an MCMM search.

TOPN — Temporary file OPeN

Opens the temporary coordinate file *filename*.tmp, where *filename* is the stem of the input file name. This command is typically used to recover the structures saved during a failed MULT or MCMM run by combining the atom connection table from the input file and atomic coordinates from the .tmp file. A typical recovery command file is:

```
filename.mae
filename-out.mae
DEBG       1
READ
TOPN     100
BGIN
TRED
WRIT
END
```

This command file opens the *filename*.mae structure file and the *filename*.tmp temporary file, reads in the first 100 coordinate sets from the temporary file, and writes the corresponding structures to the output structure file, *filename*-out.mae. Arg1 of TOPN is the number of structures to be read from the temporary file. See the end of the failed job log file to find this number; however, be aware that the last structure in the .tmp file may be corrupted, depending on the nature of the failure of the previous job.

If this operation is to be used to recover structures from a run which used the SUBS command, then the SUBS commands in the original run must be included immediately after the READ command in the command list above.

arg1 *Number of atomic coordinate sets to be read from the .tmp file*

MMOD — Maestro-macroMODel interaction

Directs MacroModel to write a *filename*-mon.mae file, where *filename* is the stem of the .com file name, for interaction with the Maestro GUI.

Note: This file conveys the structural information to Maestro so that the molecular structure displayed can updated while the job is running. Note that Maestro uses the .log file for the textual information displayed in the Monitor panel.

Note: The writing of structural files can slow MacroModel down tremendously, especially on NFS-mounted filesystems. Thus, lowering the frequency of writing, or eliminating it entirely by removing the MMOD command, is worth considering for long jobs.

arg1 *Frequency of writing to the filename*`-mon.mae` *structural data file*

1 Default. MacroModel writes to the *filename*`-mon.mae` file as frequently as it can for real-time monitoring. For long jobs, this slows MacroModel execution.

n Write the file only 1/*n* times as frequently as the default. Thus, arg1=10 would update the *filename*`-mon.mae` file 1/10 as frequently as the maximum rate.

arg2 *Recoloring of atoms in filename*`-mon.mae` *file*

0 No recoloring will be done.

1 Atoms are recolored by energy gradient, an index of strain. A color scheme is used that follows the spectrum, where red indicates the most highly strained region of the molecule and blue indicates the most relaxed regions.

4.2 Selection of Force Field, Nonbonded Cutoffs, and Solvation Treatment

`FFLD` — Force FieLD Selection

Specifies a force field; there is no default for this command. A force field must be specified for energetic calculations.

The force-field files we supply have names *fieldname*`.fld`, where *fieldname* is one of: `mm2`, `mm3`, `amber`, `amber94`, `oplsa`, `f10` or `f11`. MacroModel first looks in the local directory for the specified force-field file and, if it is not found there, looks in the `~/.schrodinger/macromodel` directory. If MacroModel does not find these files in either of these two locations, it finally searches in the `$SCHRODINGER/mmshare-v`*version*`/data` directory. Before looking for *fieldname*`.fld`, MacroModel looks locally for a file called *filename*`.`*fieldname* (e.g., `my_job.amber`). Under certain circumstances, other processes preparing jobs for running by MacroModel use this convention.

arg1 *Force field*

1 MM2*. Allinger's 1987 parameter set with many additions [1, 2]. Used for simple organics. Differs from the authentic field by use of a Coulomb's law treatment of electrostatics and torsional barrier treatment of conjugation.

2 MM3*. Allinger's 1990 parameter set with additions [3]. Used for simple organics. Differs from the authentic field by use of a Coulomb's law treatment of electrostatics and torsional barrier treatment of conjugation.

3 AMBER*. Kollman's united atom and all atom fields with additional parameters for organic functionality [4, 5]. Used primarily for biopolymers.

4 AMBER94. Kollman's 1994 version of AMBER, Amber4.1 [6].

5 OPLSA*. Jorgensen's nonbonded parameter set + AMBER bonded functions for liquid simulations [7]. Used primarily for peptides. Best for relatively rigid molecules, because the torsional parameters have not been optimized to reproduce conformational energy differences.

10 MMFF94 and MMFF94s [8–14]. See also the description of arg4 of this command.

11 OPLS_2001. This force field, developed by Professor W. Jorgenson of Yale University, is probably the best one available by default in MacroModel for condensed-phase simulations of peptides [15]. Formerly known as OPLS-AA.

A value of 10 or 11 for arg1 encodes a force-field whose parameters are obtained from a coprocess interacting with MacroModel using the BMFF mechanism. See Appendix E for details. The force-field file name used in this case is `f10.fld` for MMFF and `f11.fld` for OPLS_2001.

14 OPLS_2005. An enhanced version of the OPLS_2001 all-atom force field developed by Schrödinger to provide a larger coverage of organic functionality. In particular all torsional parameters have been refit to reproduce the conformational energetics derived at a higher level of quantum theory and additional charges have been fit to support additional organic functionality. The parameters for proteins have been updated to the ones published more recently [52].

arg2 *Electrostatic treatment*

0 Default. Uses dielectric treatment encoded within force field file unless solvation model 3 is used (see `SOLV` command), in which case the constant dielectric treatment is used.

Note: All our force fields are supplied, by default, with constant dielectric electrostatics. Prior to MacroModel 6.0, AMBER*, MM2, and MM3 used distance-dependent dielectric constant by default.

–1 Turns Coulombic molecular electrostatics off.

1 Gives constant dielectric electrostatics.

2 Gives distance-dependent dielectric electrostatics.

arg3 *Hydrogen bonding treatment*

0 Uses equation selected in force field file (default).

1 Turns off explicit 10,12 hydrogen bonding function and uses 6,12 Lennard Jones instead.

2 Gives explicit 10,12 hydrogen bonding function.

arg4 *BMFF force-field option*

For a BMFF force-field, this requests a special option given in an "Option:" line of the MacroModel force-field file. For MMFF, a "1" in this position requests that MMFFs parameters be used; these enforce planarity about delocalized sp2 nitrogens.

arg5 *Molecular dielectric constant*

Default = 1.0. The solvent dielectric constant is normally read from the solvent file (see `SOLV` command) and should not generally be set here.

arg7 *Planarity of aromatic rings*

Enforce planarity of aromatic rings for OPLS_200*X* force fields.

0 Do not enforce planarity of aromatic rings. This is the default.

1 Enforce planarity of aromatic rings.

arg8 *Coefficient for torsional damping*

Default: 100.

For highly strained conformations in which bond angles become straight, related dihedral angles can become undefined. To avoid this problem and the physically rapidly varying dihedral angle potentials for states close to this, the torsional potential is damped using the function:

$$f_{damp} = (1 - \exp(-c_{tors}(1-\cos^2\theta_1))) (1 - \exp-c_{tors}(1-\cos^2\theta_2)))$$

where θ_1 and θ_2 are the bond angles formed by the first and last three atoms in the torsional potentials.

The default value of the torsional damping coefficient, c_{tors}, of 100 damps the dihedral angle potential significantly for bond angles less than 10 degrees and works well. This argument permits one to change the value of c_{tors}. Increasing c_{tors} dramatically reduces the range over which the damping is applied.

FFOP — Force Field OPtion selection

Specifies alternative (ALT) selections that override those in the force field file. May be used, for example, to select different Z0 atom definitions. See Appendix D for details. An FFOP command must come before an FFLD command

arg1 *ALT number*

The alternative being selected in the force field file.

arg2 *ALT selection*

Number corresponding to the ASCII character defining the selection desired for the alternative given in arg1. The numbers corresponding to the various ASCII characters are given in Table 4.1

Table 4.1. ASCII to decimal number conversion table.

Char	Num	Char	Num	Char	Num	Char	Num	Char	Num
)	41	=	61	Q	81	e	101	y	121
*	42	>	62	R	82	f	102	z	122
+	43	?	63	S	83	g	103	{	123
,	44	@	64	T	84	h	104	|	124
-	45	A	65	U	85	i	105	}	125
.	46	B	66	V	86	j	106	~	126
/	47	C	67	W	87	k	107		
0	48	D	68	X	88	l	108		
1	49	E	69	Y	89	m	109		
2	50	F	70	Z	90	n	110		
3	51	G	71	[	91	o	111		
4	52	H	72	\	92	p	112		
5	53	I	73	]	93	q	113		
6	54	J	74	^	94	r	114		
7	55	K	75	_	95	s	115		
8	56	L	76	`	96	t	116		
9	57	M	77	a	97	u	117		

Table 4.1. ASCII to decimal number conversion table. (Continued)

Char	Num	Char	Num	Char	Num	Char	Num	Char	Num
:	58	N	78	b	98	v	118		
;	59	O	79	c	99	w	119		
<	60	P	80	d	100	x	120		

EXNB — use EXtended noNBonded cutoffs

Despite its name, this command can be used to specify short as well as long cutoffs.

Extended cutoff distances are, by default, 8 Å in vdW, 20 Å in charge/charge electrostatics. Standard defaults in the absence of this command are 7 Å for vdW and 12 Å for charge/charge electrostatics. Other cutoffs may be selected by adding values for arg5-8. Calculations dealing with ions should use the EXNB option.

Large distance values for cutoffs generally slow calculations but often make convergence smoother. Occasional problems with energies and gradients which appear to increase upon repeated minimizations may usually be solved by using long van der Waals and electrostatic cutoff distances. The native MM2/MM3 and MMFF programs use complete pair lists (no cutoffs) for van der Waals and electrostatic interactions.

arg 1 *Long-range derivative update interval*

Iterations (timesteps) per recalculation of long range (>5 Å) nonbonded derivatives (Default: 10), except when one of the following special values is used:

1 The constant long-range derivative option is turned off, and the entire pair list is used in the evaluation of nonbonded derivatives.

2 All nonbonded pairs are put on the pair list, and the pair list is never updated. Long-range derivatives are used.

3 Like a combination of 1 and 2: all nonbonded pairs are put on the pair list, the pair list is never updated, and the entire pair list is used in each evaluation of nonbonded derivatives.

arg 2 *Long-range derivative distance cutoff*

Distance (integer, in angstroms) for distinction between close- and long-range nonbonded interactions, used in constant derivative option. Default value is 5. Longer distances give more accurate derivatives but slow the calculation.

arg5 *van der Waals cutoff*

arg6 *Coulombic electrostatic cutoff*

arg7 *Hydrogen bonding cutoff*

Default: 4.0 Å. There is usually no reason to change this.

arg8 *Fmm cutoff*

Any fixed or frozen atoms are placed within this distance of a moving (SUBS) atom in a special class called Fmm. The Fmm atoms are treated in greater detail than the other fixed or frozen atoms in GB computations. All fixed and frozen atoms have their GB radii properly recomputed when the moving atoms move. When the polarization free-energy (Gpol) is computed, fixed-moving interactions are computed using these updated GB radii. However, the GB contributions from pairs of fixed or frozen atoms utilize the updated GB radii only when both fixed atoms are Fmm. Experiments performed so far have indicated that a reasonable value for CutFmm is 8.0 Å, and this is the default.

Regardless of the setting of arg8, the program does not allow the Fmm cutoff to exceed the larger of the van der Waals and the electrostatic cutoff.

0 Default: 8.0 Å

>0 Interpreted as cutoff in Å.

<0 Interpreted as 0.

EXN2 — EXteNds EXNB

This sets CutsFm, the assumed maximum distance in angstroms that two atoms can be from each other and still influence each others' solvent-exposed surface areas. This cutoff is used for one-time computations of solvent-exposed surfaces with fixed or frozen atoms.

arg 5 *CutsFm cutoff*

0 8.0 Angstroms (default).

n *n* Angstroms.

BDCO — Bond Dipole CutOffs

This option turns on the use of Bond Dipole Cutoffs for the truncation of electrostatic and GB interactions. Two parameters can be specified: the cutoff distance for charge-dipole interactions and the cutoff distance for charge-charge interactions. The cutoff distance for dipole-dipole interactions is taken to be EXNB arg6. If EXNB is not specified, the default value for electrostatic cutoff distance is used (see the EXNB opcode description on page 37).

Limitations:

- If the EXNB opcode is used in the .com file, it *must* precede the BDCO opcode.
- Use of the BDCO opcode automatically turns on DEBG flags 89 and 90. BDCO treats all atoms, including hydrogens, explicitly for the purposes of generalized Born solvation. However, if the file structure itself employs united atoms, such atoms are treated as united atoms. In other words, all atoms in the structure file are treated as they are for the purposes of generalized Born solvation in conjunction with BDCO.

arg5 Cutoff distance for charge-dipole interactions

The default value is *sqrt(cutes**3)* where *cutes* is the electrostatic cutoff distance as per EXNB.

arg6 Cutoff distance for charge-charge interactions

The default value is 9999999.0, which effectively includes all such interactions.

These default settings typically yield shorter nonbonded pairlists than using residue-based cutoffs and the same value for EXNB arg6.

Related DEBG flags: 121, 126, 127, 128, 129.

SOLV — SOLVation selection

Specifies a solvation model (arg1) and a solvent (arg2), so that energy calculations include the approximate effects of solvent.

Solvation model 1 involved explicit solvent and is no longer supported.

Models 2 and 3 read the appropriate solvent file named *solvent_name*.slv. The file water.slv is available for models 2 and 3, while chcl3.slv and octanol.slv are available for model 3. MacroModel first looks in the local directory for the .slv file and, if it is not found, looks in the $SCHRODINGER/mmshare-v*version*/data directory.

Solvent model 2 (arg1 = 2) is purely a surface-area-based model. We recommend model 3 for all computations where solvation energies are desired. Model 2 operates as described by Hasel et al. [16]. See also Ooi et al. [17] for the parameter set given in water.slv. Solvent model 3 provides a volume-based continuum model (the GB/SA model) for the electrostatic (polarization) component. [18]

Using model 3, molecular electrostatics should be carried out with a constant dielectric treatment and a low molecular dielectric constant (e.g., 1.0). Constant dielectric electrostatics will be set automatically whenever solvent model 3 is used regardless of the default electrostatic equation selection in the force field file. EXNB should also be used with solvent model 3.

Parametrizations specific for OPLS-AA and MMFF(s) have been added to the `water.slv` file for model 3. Other force fields continue to use the default parametrization which is unmodified from previous releases. The default parametrization in the `octanol.slv` file was constructed for the MMFF force field. As well, a parametrization specific for OPLS-AA is present in the `octanol.slv` file. The new parameters for MMFF are based on the parametrization described in [19] for water, and in [20] for octanol. We gratefully acknowledge C. H. Reynolds's assistance in utilizing these parameterizations. Available force field specific parameterizations are used by default unless the program is instructed otherwise (see arg2 below).

Calculations with solvation use periodically updated constant area and/or polarization derivatives to speed the calculation. Default update frequencies are given below. These frequencies can be changed via arg3 and arg4. If difficulties in achieving low gradients are found or if dynamics in solvent is unstable, reduce these numbers (e.g., to 2). Energy minimizations and molecular dynamics simulations using continuum solvation models 2 and 3 run approximately 1/2–1/4 the rate of *in vacuo* calculations.

Note: MacroModel carries out energy minimizations with an analytical, approximate function for surface areas. Thus, intermediate energies reflect the approximate function. The final energies reported, however, use an accurate numerical function. Thus, intermediate and final energies will differ.

It is recommended that the `EXNB` opcode be used in conjunction with solvent model 3 (GB/SA) and that the electrostatic cutoff distance be set to 20 Angstroms.

arg1 *Solvation model*

2 Total solvation based on approximate solvent accessible surface areas (Scheraga's parameters).

3 GB/SA Solvation Model. Cavity and Van der Waals components from approximate solvent accessible surface areas, and electrostatic (polarization) component from GB mode. See Still et al. [18] for a discussion of effective Born radii calculation. This is the best solvent model to use.

arg2 *Solvent*

If arg2 < 0 use the general parameterization from the `.slv` file rather than the parametrization for the specific force field.

1, –1 Water (models 2 and 3)

5, –5 $CHCl_3$ (model 3)

9, –9 Octanol (model 3)

arg3 *Surface-area derivative update frequency*

Note: This is used in models 2 and 3. Default: 25 (new value in MacroModel 6.5) for PRCG, SD, and OSVM minimization modes and molecular dynamics, and 1 for FMNR and TNCG. Set to 1 or 2 for problem minimizations using SD, PRCG or OSVM; the default value cannot be overridden for FMNR and TNCG.

arg4 *Solvent polarization reset frequency*

Uses constant long range components except during resets. Default: 10.

arg5 *Minimum solvent/solute distance*

Used to remove overlapping solvent molecules in model 1. (Default: 2.5 Å)

arg6 *Flat-bottom positional constraint force constant*

Used for restraining solvent molecules in model 1. (Default: 10. kcal/mol-Å^2)

arg7 *Half-width of flat-bottomed positional constraint*

Used for restraining solvent molecules in model 1. (Default: 2.5 Å)

arg8 *Maximum distance from solute centroid*

Used in model 1. Molecules beyond this distance will have flat-bottomed constraints as defined in arg6 and arg7. (Default 0.0 Å)

`LOGP` — LOG P partition coefficient estimation

This command instructs the program to estimate the logarithm of the partition coefficient between two solvents, log $P_{\text{solv1,solv2}}$, for a series of molecules using the relationship:

$$\log P_{\text{solv1,solv2}} = (\Delta G_{\text{solv2}} - \Delta G_{\text{solv1}})/(2.30RT)$$

where $P_{\text{solv1,solv2}}$ = [Solute]$_{\text{solv1}}$ / [Solute]$_{\text{solv2}}$, ΔG_{solvN} is the free energy of solvation of the molecule in solvent N, R is the gas constant, T is the temperature in kelvin. The first solvent, solv1, must be specified by an earlier `SOLV` command. The second solvent, solv2, is specified by arg2 as described below.

`LOGP` must be followed by a `BGIN`/`END` loop containing appropriate `READ`, `AUTO`, and `MINI` opcodes. The `AUTO` opcode should set arg6 = 1.0000 for serial calculations and arg2 = −1 to avoid generation of lists of comparison atoms. Each molecule in the input file is minimized twice, once in each solvent. When `LOGP` calculation is enabled the behavior of subsequent `READ` statements is modified and alternates between reading in new molecules and switching

the solvent. Information on the free energy of solvation is collected at subsequent MINI commands. Section 7.3 of the *MacroModel User Manual* has an example of a LOGP calculation with a .com file.

The solvation models are parametrized for ambient conditions. As the temperature begins to deviate significantly from such conditions the solvation energy estimates and hence the calculated log $P_{\text{solv1,solv2}}$ become less reliable.

arg1 *LOGP on/off*

–1 Turn off log P estimation and generate a report.

0 Turn on log P estimation.

arg2 *Solvent*

Like arg2 of SOLV, if arg2 is less than 0 then use the default parametrization from the .slv file.

1, –1 Water (models 2 and 3).

5, –5 $CHCl_3$ (model 3).

9, –9 Octanol (model 3).

arg5 *Temperature*

0.0 Use the current temperature for log P calculations (initially 298.15).

> 0.0 Use this temperature for log P calculations.

Related DEBG flags: 530 and 531.

CHGF — CHarGe File

This command causes the atomic charges used in a MacroModel energy-related calculation to come from the input structure file. If CHGF is not used, then standard charges are computed according to data in the force field file.

arg1 *Source of charges*

–1 Turn CHGF off.

0 Use atomic charges from input structure file (default).

1 Same as –1.

2 Use formal charges for mono-nuclear ions from the input structure file.

3 Use formal charges for all atoms and bond orders for all bonds from the input structure file.

4 A combination of 0 and 3. Use charges, formal charges for all atoms and bond orders for all bonds from the input structure file.

arg2 *Treatment of sp3 CH_n groups in GB solvation.*

Different charge sets can be specified for Coulombic and for GB solvation calculations. The charges used for Coulombic calculations are written to the first charge column in the output file; the charges used for GB calculations are written to the second charge column.

Note: The only place in which this facility is currently used is in the treatment of sp^3 CH_n groups. When charges are assigned by the force field, then, for the purpose of GB calculations only, charges on hydrogens in such a group are added to the charge on the carbon, and the entire group is treated as a united atom.

When reading an input file, the values in the first charge column are used for Coulombic calculations and those in the second charge column are used for GB.

0 (Default.) If all H atoms of an all-atom CH_n group have zero charge, unite the group for GB calculations; otherwise, treat these H atoms explicitly. If a file is written out containing force-field charges, this default recaptures the force-field behavior should the file subsequently be read in with `CHGF` in effect.

1 Never unite all-atom sp^3 CH_n groups for GB.

2 Always unite all-atom sp^3 CH_n groups in GB. This allows a structure with equal charge columns to be read in with `CHGF` and for the default behavior to be embodied in the output; that is, the output will be suitable for reading with `CHGF` arg2=0.

4.3 Program Flow Control

BGIN — loop BeGIN

Begin a command loop for minimizing a series of structures.

arg1 *Number of passes through the* BGIN/END *loop*

0 Continue looping until some other termination condition—such as an end-of-file—is encountered.

>1 Execute this number of passes through the loop.

arg2 *Behavior if an error is encountered while in the loop*

0 Exit the program.

1 Skip to the top of the next iteration. This option is useful if one is minimizing many diverse structures, some of which may lack appropriate parameters.

2 Attempt to continue execution.

END — loop END

End a command loop. Commands between the BGIN and END will be executed repetitively. BGIN/END loops cannot be nested.

REST — RESTart

This command, if placed into a command file, will try to restart a process that was interrupted by a system crash. Note that REST cannot be used with Monte Carlo runs at this time. Structures from failed Monte Carlo runs may be retrieved using the TOPN and TRED commands (see TOPN on page 32). Retrieved structures may be appended to the results of other MC runs, then reminimized using the MULT command to give a globally unique set of conformers.

RWND — ReWiND file

Rewind the current input or output file and use it as input for subsequent commands. This has only been tested for MINI, LMCS, LMC2, and MBAE commands following the RWND.

Note: A READ command should occur after the a RWND command before commands which perform additional calculations on the structures present.

arg1 *Which file is to be rewound*

0 The current output *filename*-out.mae file.

1 The current input *filename*.mae file.

arg2 *Disposition of intermediate output*

0 Discard; at the end of program execution, there will be a single output file, typically named *filename*-out.mae, that contains only the output generated following the last RWND command.

1 Keep intermediate output in separate files, typically named *filename*.ou1, *filename*.ou2, etc. The final output appears in *filename*-out.mae.

2 Keep all intermediate output, together with final output, in a single file, typically named *filename*-out.mae.

4.4 Hydrogen Addition and Deletion

HADD — Hydrogen ADD

Add hydrogens and lone pairs to a structure. This command must come immediately after the READ command if it is used.

arg1 *Control*

0 Hydrogens added to all atoms of structure

1 Hydrogens added to all non-carbon atoms of structure

HDEL — Hydrogen DELete

Hydrogen delete, opposite of HADD. This command must come immediately after the READ command if it is used. This command deletes lone pairs and hydrogens attached to carbons only.

4.5 Energy Calculation

ELST — Energy LiSTing

Calculate energy of current structure and dump Energy LiSTing to one or more output files. Can be used within a BGIN/END loop to compute energies of all structures in a file. ELST cannot be used with the MC conformational searching commands (MCMM, MCSM).

By default, very close nonbonded atomic pairs will be separated before the energy calculation is done. To override this, specify DEBG flag 33. By default the first 100 such pairs encountered

are reported unless DEBG flag 36 is specified, then all such pairs are reported. DEBG flag 33, disables the automatic separation of close pairs.

arg1 *Extent of listing, format of the output structure file*

–1 List the total molecular mechanics energy to the log file only.

Note: For Embrace jobs using the MBAE opcode, Maestro assigns –1 as the value for ELST arg1 in the .com file, since using a different value could result in a very large .mmo file.

0 List the total molecular mechanics energy to the log file and the minimal Energy Summary to the file *filename*.mmo.

1 Write the total molecular mechanics energy to the log file and the complete Energy Listing with all internal coordinate components to *filename*.mmo.

Note: Setting ELST arg1=1 can result in a very large .mmo file. For certain types of jobs (i.e., those using MBAE), it is recommended that arg1 be set to –1. Otherwise, the .mmo file created could be very large. Alternatively, shorter cutoffs (see EXNB, EXN2, and BDCO) may be used to help keep the .mmo file to a manageable size, if you still wish to have arg1 set to 1.

2 Write double and single precision molecular mechanics energy to log file only. Used for testing or for short summary of total stretch, bend, and other energies.

3 Acts like option 0 except, in addition, numerical and analytical surface areas for all atoms are listed to the .mmo file. Surface energies are also given and are computed using the method specified in arg3.

4 Prints the double precision energy components to the .log file. In the past, this has been possible only in combination with single-precision energies (arg1=2).

arg2 *Energy units for log file listing*

Affects the energies listed in *filename*.mmo as well as the log file.

0 kJ/mol (default)

1 kcal/mol

arg3 *GB/SA solvation numerical area/Born radii options*

0 Use numerical evaluations of atomic surface areas and analytical approximation for Born radii (default).

1 Use fast analytical approximations of surface areas and Born radii.

2 Use numerical evaluations of both surface areas and Born radii.

arg4 *Updating of interaction array.*

Non-default options are useful primarily for debugging.

0 (Default.) Program uses internal criteria to decide whether to update the interaction array.

1 Update the full interaction list.

2 Update nonbondeds only.

`DLST` — Derivative LiSTing

List derivatives for individual atoms to log file. Used for testing. Listed derivatives include first and second derivatives (listings are long so use only on small molecules). When both numerical and analytical derivatives are listed, any discrepancies between analytical and numerical derivatives of more than 5% will be marked by a "*".

arg1 *Derivative selection*

1 Numerical and analytical first derivatives (default).

2 Numerical and analytical second derivatives.

3 Numerical and analytical first and second derivatives.

4 Analytical (only) first derivatives.

5 Analytical (only) first and second derivatives.

arg2 *Atom number where derivative checking is to begin*

0 Start at atom 1 and give derivatives for entire system.

>0 Start at atom arg2 and give derivatives for rest of system.

<0 Start at atom 1 and give derivatives for atoms through –arg2.

arg3 *Potential function being tested:*

0 All (default)

1 Stretch

2 Bend

3 Torsion (proper and improper)

4 Nonbonded

8 Solvation 1

9 Solvation 2

10 Solvation 3

11 Stretch-bend

12 Bend-bend

13 Stretch-torsion

14 Charge-multipole electrostatics

15 Wilson-angle out-of-plane terms

arg4 *Number of calls to derivative routine before derivative checking*

Used for testing of constant derivative options (Default: 1).

arg5 *Step length (Angstroms) for numerical derivative calculation*

Default depends on the potential function selected, but is 0.0001 except for Solvation 1 and Bend-bend.

ASET — Atom SETs

This command is used to place atoms into mutually exclusive sets. During a subsequent ELST procedure (with any ELST arg1 value other than 4) the energy components are printed separately for the interactions within each set, as well as between pairs of sets. Although the sets must be mutually exclusive, they need not, collectively, include all the atoms in the molecule. If DEBG flag 1 is turned on, the set membership is listed in the log file at the time of the ELST command.

The various ASET energies are a pairwise breakdown of the interaction energies obtained by adding up the individual atom-atom interaction energies. This approach has difficulties with GB/SA solvation because it is not a pairwise additive potential. In the generalized Born (GB) approach the solvation energy depends on the Born radii each of which depend in a complex

way on the arrangement of large number of atoms in the system. The ASET mechanism takes a static approach to calculating the GB solvation energy in which this dependence of the Born radii is ignored, leaving a simple pairwise sum of solvation terms. The surface area (SA) portion of the solvation energy calculation is more problematic because from a single conformation it is not obvious how much the surface area is affected by association. Also, because the solvent inaccessible regions are often excluded by multiple atoms, a pairwise breakdown of the energy solvation energy is problematic. The ASET mechanism takes a simple approach in that it associates the SA solvation term with the interaction of a set with itself.

ASET commands may be issued before a READ, and are in force for all subsequent structures read until or unless cleared or altered. Thus, such commands may be issued before a BGIN/END loop, and will be in force for all structures read within the loop.

Arg7 and arg8 are only used when MBAE is not used.

arg1-4 *Atom number*

The action that is applied to these atoms depends on the settings of arg5 and arg6. If arg1-4 are all zero, then the action is applied to all atoms in the system.

arg5 *Set number*

May be any non-negative integer, specified as 0.0000, 1.0000, 2.0000, etc. Only sets one through 20 will be considered during a subsequent ELST. Set 0 stands for *no set*; placing an atom in Set 0 is the equivalent of removing it from whatever set it had been in. Recall that the sets are non-overlapping: an atom is a member of only the last set into which it is placed.

arg6 *Command mode*

Should be given <an integer value (such as –1.0000). This argument is used to specify the action to be taken, as follows:

0 Add the listed atoms to the set.

1 Synonym for 0.

–1 Delete the listed atoms from the set given, if they are in the set. Attempts made to delete atoms from set 0 are ignored.

2 Add the range of atoms between arg1 and arg2 (inclusive) to the set; arg3 and arg4 are ignored.

–2 Delete each atom in the given range from the set, provided it is in the set.

3 Add the molecules containing the atoms given by args1 through 4 to the given set.

–3 Delete the molecule containing the atoms in args1 through 4 from the set, if they were in it.

arg7 *Property for inter/intra set energies*

Write inter-set or intra-set interaction energies as properties to the structure output file. Should be given an integer value (such as 2.0000).

0 Inter-set energies between set 1 and all other sets (default).

2 Inter-set energies between all sets.

4 Intra-set energies for all sets.

arg8 *Select energy components*

Select the energy components that are written as properties to the structure output file. Should be given an integer value (such as 2.0000).

0 Record only the total energy (default).

2 Record the total energy and non-bonded energy.

4 Record all energetic components.

ASNT — turn off Atom Set iNTeractions

This command allows energetic interactions between sets to be turned on and off. The sets must be defined using the ASET command. Separate control is available over force-field interactions and constraint interactions imposed using the FXDI, FXBA, and FXTA commands.

ASNT commands may be issued before a READ, and are in force for all subsequent structures read until or unless cleared or altered. Thus, such commands may be issued before a BGIN/END loop, and will be in force for all structures read within the loop.

arg1 *Set number or control*

>0 The first set number.

<0 The actions encoded in arg3 and arg4 will be applied to all pairs of sets.

arg2 *Set number or control*

>0 The second set number.

<0 The actions encoded in arg3 and arg4 will be applied pairwise to all set combinations including the set encoded in arg1, which must be positive.

arg3 *Force-field interaction control*

0 Turn off force-field interactions between sets.

1 Leave force-field interactions on.

arg4 *Constraint-interaction control*

0 Turn off constraint interactions between sets.

1 Leave constraint interactions on.

Examples:

- `ASNT` **1 2 0 0**: Turns off all interactions between sets 1 and 2.
- `ASNT` **1 2 0 1**: Turns off force-field interactions but retains constraint interactions between sets 1 and 2.
- `ASNT` **2 –1 0 0**: Turns off all interactions between set 2 and other sets.
- `ASNT` **–1 0 1 1**: Turns on all interactions between all sets (i.e., restores initial state of the program).

`VDWB` — Van Der Waals Bends

Used to model the coordination sphere of an inorganic complex, using the "points-on-a-sphere" model as described by Hay [21]. In this model, the mutual interaction of a pair of ligands bound to the same metal is handled by means of a van der Waals, rather than bond-angle-bending interaction. This command replaces specified bond-angle interactions with van der Waals interactions.

Note: Coulombic components are removed from these interactions, unless `DEBG 12` is specified.

`VDWB` may be used along with the `MCMM`, `LIGB`, and `MOLS` commands to perform a configurational search of the coordination sphere. `TORS` and, if applicable, `RCA4` commands can be added to explore the internal conformational space of the ligands at the same time.

arg1 *Central atom*

Typically, the metal atom. Bends about this atom are replaced with vdW interactions.

arg2 *Outer atom*

0 All bends with arg1 as the central atom with be replaced with vdW interactions.

>0 Arg3 must also be greater than 0. In this situation, the arg2-arg1-arg3 bond will be replaced by a vdW interaction.

<0 Arg3 must also be less than 0. In this situation, if arg1 already has some bends on the VDWB list, the ABS(arg2)-arg1-ABS(arg3) bend is removed. If no such list exists for arg1, one is created and all bends centered on arg1 are placed on it except ABS(arg2)-arg1-ABS(arg3).

arg3 *Outer atom*

See arg2 description.

arg5,6 *r_o and ε for the van der Waals pairs modeled by VDWB.*

If these are specified, all pairs will be modeled using the same parameters: the last nonzero values specified in any VDWB command. If r_o and ε are not specified, atom-type-dependent van der Waals parameters from the force-field are used. To achieve a nearly purely repulsive potential, use, for example, r_o = 9.0 and ε = 1.0E–6. This is nearly identical to the repulsive part of a MacroModel C1-C1 nonbonded interaction using the AMBER force field. Our own experience indicates that default parameters (arg5 = arg6 = 0) give good results.

4.6 Energy Minimization

In this section we describe the SDLP command, which explicitly searches for saddle-points, as well as commands associated with energy minimization.

MINI — MINImize the energy of a structure

Used after the READ command, and if used within a BGIN/END loop, will minimize the energy of all structures in the input file. If MULT is used then structures will be checked for duplication (see COMP) and any duplicates found will be eliminated from the output file. Stereochemistry is generally maintained during minimizations but strained systems can allow inversion of chirality especially if starting geometries are high in energy. If all the structures in the file are the same except for conformational differences, then CHIG commands can be used to save the stereochemistry (i.e., to reject any structure with a chiral center different from that found in the first structure in the input file).

The AUTO opcode can be used for automatic setup of MINI calculations.

If solvation is used in a minimization, the structure is minimized with analytical approximations to surface areas, but the final energies reported use accurate numerical areas.

If energies change substantially on reminimization or nonbonded updates, use the EXNB command to increase the nonbonded cutoff distances for electrostatics and van der Waals. Alternatively, update more frequently using arg7.

arg1 *Minimization mode*

<0 Allow uphill motion, using method |arg1|; but implemented only for arg1=4 using no line search (arg2= 0). This facilitates saddle-point searches.

0 Steepest Descent with or without line searching (SD)—Not generally useful except for highly strained structures. Should be used with line searching (arg2 =1).

1 PR Conjugate Gradient (PRCG)—Best general method. [22]

3 OS Variable Metric (OSVM). One of the best variable metric methods. There is no built-in limit on the number of atoms in the molecule when using OSVM. [23]

4 Full Matrix NR (FMNR)—Best method for fully converging molecules. There is no built-in limit on the number of atoms in the molecule. Gradient must be small before using this method. Use with line searching if gradient >0.5 kJ/Å-mol.

9 Truncated Newton (TNCG)—Superb method for flexible structures. There is no limit on the number of atoms in the molecule when using TNCG. [24]. See arg6.

10 Limited Broyden-Fletcher-Goldfarb-Shanno (LBFGS)—LBFGS can be specified by MINI arg1=10. Experiments so far have not shown superiority of this method above all others; however, we expect to improve the implementation as time goes on, as we feel this method has great potential.

11 LTNCG—LTNCG is virtually identical to TNCG (MINI arg1=9), but it requires less memory and thus can be used on larger molecules. Differences between TNCG and LTNCG:

TNCG is effectively single precision, but LTNCG is fully double precision.

Because of the difference between the Hessian cutoff used by TNCG and LTNCG (see ARPK arg8), and the limited memory allowed for Hessian storage in LTNCG (see ARPK arg4), the sparse Hessian representation of TNCG and LTNCG might differ. Nevertheless, LTNCG gives practically identical results with TNCG on small to medium sized molecules (up to approx. 1000 atoms).

For molecules for which both TNCG and LTNCG can be used, TNCG is faster by a factor of approx. 1.5 to 2.

20 Use TNCG method if the number of unfixed atoms is less than 1000 and solvation is employed; otherwise use PRCG. Sets arg2 to 1.

arg2 *Line searching protocol*

0 Default line searching

1 For SD or FMNR or LBFGS, turns on line searching. Although LBFGS can be used with line search, it is not recommended in the current implementation.

For PRCG, selects 3-point line searcher (use with problematic structures)

For TNCG, selects original line searcher (try for problematic structures)

arg3 *Maximum number of iterations*

The default of 0 iterations may be used when the semantics of surrounding commands requires a `MINI` command but it is not desired to actually perform a minimization. When a minimization is run from MacroModel, arg3 is set to 500 by default. For PRCG and OSVM minimization methods, some small multiple of 3*N* steps often suffices, where *N* is the number of atoms in the system being simulated.

arg4 *Energy code (see* `DEMX`*)*

arg5 *Step-size buffer*

This variable, 1.0 by default (1.1 for PRCG), is used to increase or decrease the step size during minimization. If arg5>1, then convergence may be accelerated, but the procedure can also become unstable. If arg5<1, then convergence will be slowed, but the minimization may run more smoothly with problematic structures. With normal structures, however, the default value of 1.0 should be used.

Arg5 behaves somewhat differently if FMNR is specified without line-searching (arg1=4, arg2=0). Then, by default (arg5=0.0), the algorithm takes its full step-size, which is the estimated distance to the minimum determined by a harmonic fit to the potential energy surface. This corresponds to the ordinary meaning of arg5 for other methods. This distance will be too great (for example, it may overshoot the minimum) unless one is already close to a minimum. If arg5 is nonzero for FMNR, its value is used as the maximum distance in angstroms that any atom can move in a single step. If the normal `FMNR` algorithm calls for a larger step, then all coordinate motions are scaled back by the ratio of the arg5 value to the maximum atomic displacement called for by the normal algorithm. Experimentation is usually called for to determine a good non-default value for arg5, but if the gradient at the start of

FMNR has already been reduced to less than about 10 kJ/mol·Å, a value in the range 0.1 to 1 Å is a good starting point. Provided the gradient is already reasonably low (for example, by virtue of an earlier conjugate-gradient minimization with "loose" convergence), this option permits convergence to a saddle point, since FMNR actually tends toward singularities of any sort, not just minima.

Note: This variable does not affect minimization when used for TNCG minimization (arg1=9) and the original line searcher (arg2=1).

arg6 *TNCG Hessian cutoff*

Default is 0.5 kJ $mol^{-1}Å^{-2}$. Smaller values (e.g., 0.1, 0.01) slow down the iterations but improve the convergence properties of the method.

arg7 *Maximum movement (Å) before nonbonded update*

Set to a small number (e.g., 0.1) for frequent updates. Default (0.0) gives an update frequency which is a function of the van der Waals cutoff distance but is minimally 0.5 Å.

arg8 *Reporting interval*

0 (Default.) No intermediate reporting, unless DEBG 1 has been specified.

n *n*>0 If *n* is an integer, report the energy, RMS gradient, and the RMS atomic movement every *n* steps. If *n* is not an integer, then the maximum atomic movement and the maximum absolute gradient component are reported, in addition to the reporting of energy, RMS gradient, and the RMS atomic movement. Overrides reporting interval set by DEBG 1.

CONV — minimization CONVergence criterion

The batch minimizer uses default derivative convergence criterion at a value of 0.05 kJ/Å-mol (ca 0.01 kcal/Å-mol) if no CONV command is used.

arg1 *Type of criterion*

0 Iterate until max number of iterations has been achieved.

1 Energy convergence.

2 Derivative convergence (default if no CONV record appears).

3 Movement convergence.

arg2 *Extent of convergence (kJ/Å-mol)*

0 1.0

1 0.1

n 10^{-n}

arg5 *Real number extent of convergence (kJ mol^{-1} Å^{-1})*

Overrides arg2 value.

MWRT — Minimization WRiTe

Chained minimizations (on the same structure) can be run using the MWRT opcode. Following an MWRT command with arg1=0, information on the subsequent MINI commands is stored without actually performing them. After a second MWRT command with arg1=1, the optimizations begin at the next instance of a MINI command. The program performs all the previously stored MINI commands followed by the current MINI command sequentially, until convergence is attained (as defined by the CONV opcode). If convergence is obtained before the stored MINI commands are completed, the remaining MINI commands are skipped. This capability allows you, for example, to start a minimization with a steepest-descent minimizer and end with a full matrix Newton Raphson minimizer. MWRT can be used in conformational searches or multiple minimizations. A maximum of 11 MINI commands can be chained using this opcode.

arg1 *Enable writing of MINI commands*

0 Store MINI commands without doing minimizations until arg1 =1 is encountered.

1 Run stored minimizations starting at the next MINI command.

MTST — Minimization TeST

This command computes the first and second energy derivatives for a minimized structure and returns the first derivative root-mean-square value and test for imaginary vibrational modes, indicating whether the current structure is a minimum, a saddle point or something else. The procedure operates by counting the number of frequencies less than arg5 (default = 2 cm^{-1}) beyond the lowest 6 which correspond to free translation and rotation in a minimized structure. Results are written to the log file.

Limitations:

- MTST cannot be used during a MULT run.
- MTST can only be used after a MINI command, unless DEBG 211 is used and MTST follows an ELST command.

arg1 *Listing of vibrational frequencies*

0 No listing.

1 Listing to `.mmo` file.

arg5 *Lower limit for frequency reporting*

The lowest vibrational frequency considered to be a real vibrational mode (default 2.0 cm^{-1}). Note that the 6 lowest frequencies are always skipped.

`VIBR` — visualize VIBRational modes

This command allows for the visualization of molecular vibrations. A file is created which can be read by the ePlayer facility in Maestro. See Section 8.6 of the *Maestro User Manual* for instructions on using the ePlayer. For each mode in the file, the minimized structure comes first; then there are some number of frames (call it *N*) exploring the mode in one direction, 2*N* frames exploring it back in the other direction, and finally *N* additional frames exploring it "forward" again. Each mode is depicted in a different color (up to five). The trivial modes (determined by frequency, not counting) are skipped.

This command is allowed only if a `MINI` has been performed, unless `DEBG 211` has been set.

arg1 *First mode to animate*

0 1 (default). Animate starting with the first mode.

n For *n*>0, use this value as the first mode to animate instead of the default. Animate starting with the *n*th mode.

arg2 *Last mode to animate*

0 Only the mode specified by arg1 will be animated.

n For *n*>0, use this value as the last mode to animate. Animate the range of modes specified by arg1 and arg2.

arg3 *Number of frames for 1/4 period*

This number corresponds to *N* in the above command description.

0 10 (default).

N For *N*>0, use this value instead of default.

arg4 *Print level*

0 Default: Only eigenvalues are printed in the `.log` file.

Eigenvalues and eigenvectors are printed in the `.log` file. Note that printing of eigenvectors generates copious output. This is only recommended as a debugging tool.

arg5 *Amplitude (Å)*

Distance the fastest-moving atom will move in the first *N* frames. Note that this has the same definition as the step size in `LMCS`.

0 1.0 (default).

>0 Use this value instead of the default.

`VBR2` — visualize ViBRational modes, 2nd version

This command allows for the visualization of molecular vibrations. A file is created which can be read by the ePlayer facility in Maestro. See Section 8.6 of the *Maestro User Manual* for instructions on using the ePlayer. For each mode in the file, the minimized structure comes first; then there are some number of frames (call it *N*) exploring the mode in one direction, 2*N* frames exploring it back in the other direction, and finally *N* additional frames exploring it "forward" again. Each mode is depicted in a different color (up to five). The trivial modes (determined by frequency, not counting) are skipped.

`VBR2` is virtually identical to the `VIBR` command, but `VBR2` uses ARPACK and can be applied to a molecule of any size. There are two small differences:

- The default amplitude is set to 5 Å in `VBR2` in arg5 instead of 1 Å.
- `VBR2` also uses arg6, to specify a minimum eigenvalue, below which modes are considered to be trivial (ro-translational) and they are ignored.

arg1 *First mode to animate*

0 1 (default). Animate starting with the first mode.

n For *n*>0, use this value as the first mode to animate instead of the default. Animate starting with the *n*th mode.

arg2 *Last mode to animate*

0 Only the mode specified by arg1 will be animated.

n For *n*>0, use this value as the last mode to animate. Animate the range of modes specified by arg1 and arg2.

arg3 *Number of frames for 1/4 period*

This number corresponds to *N* in the above command description.

0 10 (default).

>0 Use this value instead of the default.

arg4 *Print level*

0 Default: Only eigenvalues are printed in the `.log` file.

Eigenvalues and eigenvectors are printed in the `.log` file. Note that printing of eigenvectors generates copious output. This is only recommended as a debugging tool.

arg5 *Amplitude (Å)*

Distance the fastest-moving atom will move in the first *N* frames. Note that this has the same definition as the step size in `LMCS`.

0 5.0 (default).

>0 Use this value instead of the default.

arg6 *Minimum magnitude of eigenvalues of real vibrational modes*

0 Default: 0.001.

>0 Other value to be used.

`RRHO` — RRHO normal mode analysis

`RRHO` stands for rigid-rotor, harmonic oscillator calculation of translational, rotational, and vibrational enthalpy, heat capacity, entropy, partition function, etc. These properties are written to the log file and to the output structure file as structure-level properties.

The vibrational calculation ignores vibrational frequencies below arg7 cm^{-1} (default 2.0). The rotational and translational calculations ignore the effects of fixing or freezing atoms. The vibrational contributions are affected by fixing or freezing. If no atoms are fixed or frozen, for a true minimum energy structure, there should be 6 such ignored frequencies in the range of –1.0 to 1.0 cm^{-1} corresponding to free translation and rotation. Any frequency substantially

more negative indicates a maximum of some sort (e.g., a saddle point). We advise listing the vibrational frequencies (MTST command) to be sure that arg7 is appropriately set if there are more than 6 skipped frequencies or if there are any large negative (imaginary) frequencies.

If there are 3 or more noncollinear fixed or frozen atoms, net rotations and translations are not permitted and the first 6 modes correspond to either intrasystem vibrations or, in the case of fixed atoms, mixed intrasystem/restraining force vibrations. If frozen atoms are used and fixed atoms are not then the first 6 modes are purely intrasystem and thus should contriubute to the vibrational terms. In such cases arg4 should be set to a value less than 0 to instruct the program not to automatically skip the first 6 vibrational modes (unless they are less than arg7 or 2 cm^{-1} if arg7 is 0).

RRHO can only be used after a MINI or ELST command. RRHO cannot be used with the MULT option or during a Monte Carlo conformational search.

The command can be used only on fully minimized structures having MAXFV or fewer atoms if vibration is being considered. For a discussion of MAXFV, see Section 1.4 on page 9.

arg1 *Print mode*

0 Summary of S, C_v, and G to log file (default).

1 Detailed listing of thermodynamic parameters to .mmo file and summary of S, C_v, and G to log file.

arg2 *Calculation mode*

0 Translation, rotation, and vibration (default)

1 Rotation and vibration only

2 Translation and rotation only

3 Translation and vibration only

4 Translation only

5 Rotation only

6 Vibration only

arg3 *Symmetry number (default = 1)*

arg4 *Control over automatic mode elimination*

0 Automatically skip the first 6 vibrarional modes when calculating the vibrational contributions.

< 0 Do not automatically skip the first 6 vibrational modes. This value should be used if the system consists of frozen atoms plus unrestrainted atoms.

arg5 *Temperature (K) (default = 300.0)*

arg6 *Volume (liters) (default = 1.0)*

This parameter defines the standard state (1 molar by default) for translational entropy calculations. Use of 22.4 gives a standard state corresponding to 1 atmosphere of pressure.

arg7 *Lower limit for frequency reporting*

Lowest vibrational frequency considered to be a real vibrational mode (default 2.0 cm^{-1}). The six lowest frequencies are always skipped regardless of this setting.

SDLP — SaDdLe Point search

This is an implementation of mode-following saddle point search. For a detailed discussion of this topic, see Culot et al. [25]. This is the algorithm that initiated the development of the LMCS conformational search procedure. The basic idea of mode-following is to move uphill on the potential energy surface along a ravine starting at a minimum, toward a saddle point. The mathematical definition of a ravine is the so-called minimum energy path along which one degree of freedom is maximized while all the remaining degrees of freedom are minimized. The algorithmic implementation of mode-following is based on a coordinate transformation applied to the FMNR (MINI command, arg1=4) algorithm.

It can be shown that in a local coordinate system defined by the eigenvectors of the current Hessian during FMNR optimization, FMNR maximizes the energy along the eigenvectors with negative eigenvalues and minimizes the energy along the eigenvectors with positive eigenvalues. Mode-following is initiated by a short move along a selected low mode of the Hessian of a minimum-energy conformation. At the new point (slightly higher than the minimum energy point) the Hessian is reevaluated and its eigenvectors are calculated. The eigenvector which is most similar to the starting low-mode eigenvector, i.e., which has the largest overlap with it, is selected as the degree of freedom to be maximized.

Maximization is accomplished by taking a short FMNR step in the local coordinate system defined by the eigenvectors of the new Hessian, but following the selected eigenvector in the reverse (uphill) direction. This procedure is continued, iteratively, always following the ravine eigenvector uphill while following the remaining eigenvalues in their normal directions, until convergence to a saddle point is achieved. `SDLP` can follow multiple modes, and each mode is followed in both directions. `DEBG 94` saves all the intermediate structures in the output structure file and colors each mode-following sequence differently for better visualization.

This command is allowed only if a `MINI` has been performed, unless `DEBG 211` has been set.

arg1 *First mode to follow*

0 Default: 1 (i.e., the eigenvector of lowest frequency).

n>0 Start with the *n*th eigenvector.

arg2 *Last mode to follow*

0 Default: 1 (i.e., only follow a single mode).

n>0 Follow *n* modes.

arg3 *Maximum steps*

0 Default: 50 steps.

>0 Maximum steps for saddle point search.

arg5 *Maximum energy increase*

0 Default: 100 kJ/mol.

>0 Maximum energy increase [kJ/mol] above the starting energy minimum allowed during saddle point search. The search is aborted if this limit is exceeded.

arg6 *Maximum coordinate movement*

Maximum allowable change in any coordinate of the moving atoms. If the “natural” FMNR move would exceed this value, we would instead travel along the specified direction (eigenvector) only far enough to achieve this value.

0 Default: 0.1 Å.

>0 Maximum allowed motion in Å.

4.7 Constrained Energy Minimization

DRIV — carry out a geometry DRIVe

When included in a BGIN/END loop the DRIV command drives the specified geometric parameter through a set of fixed values. The parameter can be a dihedral angle, a bond angle, or a bond distance. The DRIV command should be followed by a MINI (not an ELST) to evaluate the energy. The driving process involves adjusting the specified geometric parameter to the starting value and then setting a constraint with a large force constant (1000 kJ/mol) for that parameter. Then an energy minimization is carried out, and in the next pass through the BGIN/END loop the parameter is incremented by the value of arg7 and the process repeated. When the final value of the parameter (arg6) is reached or surpassed then a "grid" of energy values is written to a .grd file. This grid can be displayed as a contour map in Maestro (see Section 8.3 of the *MacroModel User Manual*); its format is described in Appendix F.

You can have two consecutive DRIV commands to select two geometric parameters. You must specify the same type of parameter for each command: one angle and one distance is not an allowed combination. The inner loop runs over the parameter specified by the second DRIV command. Two DRIV commands are needed to calculate data for a Ramachandran-type plot.

Note: This opcode is incompatible with the use of DEBG 23.

arg1-4 *Atom numbers*

These define the geometric parameters to be driven. They must be valid atom numbers in the *filename*.mae file For dihedral angles they must also define an angle that is rotatable, i.e., not one in which the four atoms lie in a ring. The geometric parameter is determined as follows:

- If arg3 and arg4 are zero, the distance defined by arg1 and arg2 is driven.
- If arg4 is zero, the angle defined by arg1, arg2 and arg3 is driven.
- If arg1–arg4 are non-zero, the dihedral angle defined by arg1–arg4 is driven.

arg5 *Starting value*

This does not need to be the current value in the *filename*.mae file as MacroModel will first set the parameter to the required value. Must be nonzero for distance driving.

arg6 *Final value*

The drive finishes when the parameter is greater than or equal to this value if arg7 is positive, or when the parameter is less than or equal to this value if arg7 is negative.

arg7 *Increment value*

The increment must be nonzero. If arg7 is greater than zero then arg5 must be smaller than arg6, otherwise the drive would never terminate. Similarly, if arg7 is less than zero, arg5 must be greater than arg6.

The actual number of steps is given by the formula:

N= ABS(arg5–arg6)/arg7 +1

The maximum allowable value of N is 100. In the usual use of the `DRIV` command (i.e., two `DRIV` commands in a file), N^2 energy minimizations are done.

arg8 *Constraint Force Constant*

0 Use the default value of 1000

>0 Use this value

Units for the force constant vary with type of drive: kJ mol^{-1} $Å^{-2}$ for distances, kJ mol^{-1} $radian^{-2}$ for angles, and kJ mol^{-1} for torsions.

SUBS — define a SUBStructure for subsequent minimization

Arg1-4 are atom numbers of the substructure: those atoms that are allowed to move without restriction in the environment of anchored atoms. This command is normally used with `FXAT` commands, which restrain the position of atoms at the periphery of the substructure. The substructure and associated `FXAT` commands are commonly produced using the substructure editor of Maestro. There are two principal reasons to do this:

- To “tether” part of a molecule in space. This is ordinarily done using `FXAT` commands but no `SUBS` commands.
- To save computing time by ignoring part of the system believed to be irrelevant. If both `SUBS` and `FXAT` commands appear, then atoms that are not specified in either command are completely ignored in the simulation.

Note: When `SUBS` and `FXAT` commands appear together, any interaction except a stretch consisting only of `FXAT` atoms is eliminated, provided the following conditions hold:

- The `FXAT` atoms in question are not specified by any `SUBS` command.
- The flat-bottom half-width is zero for all the `FXAT` atoms concerned.
- Neither `DEBG 17` nor `DEBG 28` is specified.

The elimination of these interactions saves time, since `FXAT` atoms will not move much; however, stretches are always included since without them the “fixed” atoms tend to adopt unrealistic relative positions even for rather high tethering potentials.

If a SUBS command appears in the .com file with a zero-valued first argument, then MacroModel looks for a file named *filename*.sbc, in which it expects to find SUBS, FXAT, ASL1, ASL2, and ASL3 commands. This file can be produced either manually or using Maestro. ASL1, ASL2 and ASL3 commands must be added manually. Additional SUBS commands with atom numbers can be used to direct additional atoms to be added to the set in the .sbc file.

A typical use of both SUBS formats simultaneously might be simulating the binding of a large molecule to a list of small molecules. The .sbc file would define the substructure and fixed atoms for the large molecule. The first SUBS command in the .com file would have a zero-valued first argument and would set up the substructure for the large molecule. Subsequent SUBS commands would include the small molecules in the simulation. The input data files for each job would contain the large molecule first, ensuring that its numbering scheme, like that in the .sbc file, remains constant over the series of runs.

If a negative number is given for any of arg1 through arg4, then the entire molecule containing that atom number is defined as a part of the substructure. For MBAE calculations a .sbc file should be used to specify the substructure.

SUBS commands with nonzero first arguments must come after READ commands. SUBS commands with zero first arguments must come before READ, because the .sbc file is read during structure file reads.

arg1 *Substructure atom number*

0 Obtain information from .sbc file; arg3 and arg4 are ignored.

arg2 *File for substructure specification*

1 Use *jobname*.sbc.

≠1 Use *current_rootname*.sbc. For example, if a single rewind has been done then *current_rootname* is *jobname*-out.

>0 Atom number.

arg2-4 *Substructure atom number*

>0 Atom number.

arg5 *Add/Delete atoms*

0 Default, add the atoms listed to the substructure.

−1 Remove the atoms listed from the substructure.

ASL1 — ASL expression for substructure

Specify ASL syntax for a substructure. This command is intended to make it easier to set up standardized substructures that can be reused in different calculations on different molecular systems.

Note: This special command can only be used in substructure (`.sbc`) files: it cannot be used in command (`.com`) files.

The format of this command is similar to that of regular opcodes, starting with a 4-character string. The first argument is a floating-point number that has the same meaning as arg5 of `SUBS`, and the second argument is a string.

arg1 *Add/Delete atoms*

0 Add the atoms defined by the ASL expression to the substructure. (Default)

−1 Remove the atoms defined by the ASL expression from the substructure.

arg2 *ASL expression*

See the *Maestro Command Reference Manual* for information on ASL syntax.

ASL2 — ASL expression for fixed and frozen atoms

Specify ASL syntax for fixed and frozen atoms, similar to the `FXAT` opcode. This command is intended to make it easier to set up standardized substructures that can be reused in different calculations on different molecular systems.

Note: This special command can only be used in substructure (`.sbc`) files: it cannot be used in command (`.com`) files.

The format of this command is similar to that of regular opcodes, starting with a 4-character string. The first argument is a floating-point number, and the second argument is a string.

arg1 *Fix or freeze atoms*

>0 Specify the force constant for the atoms defined by the ASL expression, which are “fixed” atoms

−1 Freeze the atoms defined by the ASL expression.

arg2 *ASL expression*

See the *Maestro Command Reference Manual* for information on ASL syntax.

ASL3 — ASL expression for fixed and frozen ligand atoms in Embrace calculations

Specify ASL syntax for fixed and frozen ligand atoms in Embrace calculations, similar to the FXAT opcode. This command is intended to make it easier to set up standardized substructures that can be reused in different calculations on different molecular systems.

Note: This special command can only be used in substructure (.sbc) files: it cannot be used in command (.com) files.

The format of this command is similar to that of regular opcodes, starting with a 4-character string. The first argument is a floating-point number, and the second argument is a string.

arg1 *Fix or freeze atoms*

>0 Specify the force constant for the atoms defined by the ASL expression, which are "fixed" atoms

−1 Freeze the atoms defined by the ASL expression.

arg2 *ASL expression*

See the *Maestro Command Reference Manual* for information on ASL syntax.

FXAT — FiX ATom

Fix or freeze the position of the atom given in arg1. We use the term "fix" to mean "tether in place using a constraint"; thus "fixed" atoms can move. We use the term "frozen" to denote "rigidly freeze in place." Frozen atoms cannot move at all.

This command is generally used with the SUBS command to fix or freeze the positions of certain atoms at the periphery of the substructure being minimized. See the description of the SUBS command for a description.

When doing substructure (SUBS) calculations with FXAT restraints having no free flat bottom region (arg4 = 0), interactions wholly involving the fixed atoms, other than stretches, are eliminated from the interaction array unless DEBG 17 or 28 is set. This is primarily to improve the speed of the computation; when the force constant is large, as it is by default, the FXAT constraints, together with the stretch interactions, maintain reasonable local geometries.

The constraint potential, E_c, for a fixed atom is calculated using the equation

$$\begin{aligned} E_c &= 0 && r \le \sigma \\ &= k(r-\sigma)^2 && r > \sigma \end{aligned}$$

where $r = |\boldsymbol{R} - \boldsymbol{R}_0|$, $\boldsymbol{R}$ is the current position and $\boldsymbol{R}_0$ is the desired position (arg6-arg8), k is the force constant (arg5), and σ is the half-width of the flat-bottomed potential (arg4).

When using low force constants to "gently" constrain atomic positions, local geometries become unreasonable unless debug switch 17 is set. Also, unless mutual interactions between FXAT atoms are active, solvation energies for systems involving FXAT have no absolute meaning, although comparisons between conformers are still meaningful.

When using FXAT commands in molecular dynamics (e.g., doing substructure molecular dynamics), it is appropriate to use substantially reduced force constants (arg5, e.g., 50-100) so that realistic flexibility and rapid thermal equilibration is possible.

If two FXAT commands specify the same atom, the second replaces the first.

A FXAT command must come after READ commands.

Note that for MBAE calculations, FXAT commands should be placed in a .sbc file.

arg1 *Fixed atom number*

0 Clear or modify fixed and/or frozen atom constraints, depending on the values of the other arguments. A typical use for this facility is in homology modeling, when one might want to perform a minimization with frozen or "tight" fixed constraints, then repetitively diminish the constraints and reminimize. When arg1 is 0, the other args are interpreted as follows.

If arg2 is 0, all fixed and frozen atoms become unconstrained. If arg2 is positive, then all fixed atoms are affected according to the values of arg4 and arg5; if arg2 is negative, then all frozen atoms are so affected. For arg2 nonzero, then, if arg4 and arg5 are zero, all fixed or frozen atoms are simply unconstrained; if either arg4 or arg5 is nonzero, then the behavior is as follows.

If arg2 is positive, fixed-atom constraints are modified. If arg5 is negative, all fixed atoms are frozen. Otherwise, if arg5 is positive, all fixed atoms have their force constants multiplied by arg5 (e.g., arg5=0.1 lowers the fixed-atom force constants to 1/10 their current values). If arg4 is 0, the current flat-bottom half widths are unmodified; if arg4 is negative, any flat bottoms are removed; if arg4 is positive, then the current flat-bottom half widths are multiplied by arg4/10 (e.g., arg4=1 lowers the flat-bottom half widths to 1/10 their current values).

If arg2 is negative, then frozen-atom constraints are modified. If arg5 is positive, all frozen atoms become fixed, instead, with a force constant of arg5. If arg4 is positive, they receive a flat-bottom half width of arg4/10.

>0 Specify fixed or frozen status for the atom specified; this can override a previous specification for the same atom.

arg4 *Half width of a flat bottom restraint*

Specified in tenths of an Angstrom (an integer).

arg5 *Force constant (kJ mol^{-1} Å^{-2})*

The units of the force constant are configurable by means of the `FIX` multiplier in the force-field file in use. Internally, the program uses kJ mol^{-1} as its energy unit, so that, after multiplication by the multiplier, `FXAT` force constants have these units. Most force-field files we supply specify a `FIX` multiplier of 4.184. Thus, the units given in the `.com` file, and in the Maestro interface, correspond to kcal/mol for these force fields. In other words, the default versions of the force field files for AMBER*, MM2*, MM3*, OPLS, and MMFF indicate that fixed-atom constraints have the units of kcal mol^{-1} Å^{-2}, but the units of fixed-atom constraints for the OPLS_2001 force field are kJ mol^{-1} Å^{-2}.

0 500.0 (default).

>0 Use this value instead of default.

<0 "Freeze" this atom—make it completely unmovable. All interactions (including stretches) composed of such atoms are removed, unless `DEBG 17` is specified. Movable atoms still feel the effect of such atoms.

arg6-8 *Desired X,Y,Z coordinates of atom*

If all zero, the atom is fixed at its starting coordinates.

`FXDI` — FiX DIstance

This command supplies energetic restraints to interatomic distances. This command provides flat-bottom energetic restraint wells, in which there is no penalty for limited user-defined deviations from the equilibrium distance. This command is useful for restraining energy minimizations and molecular dynamics simulations to geometries with desired internuclear proximities (e.g., from NOE experiments).

The constraint potential, E_c, for a fixed distance is calculated using the equation

$$E_c = kd^2/2$$

where

$$d = 0 \qquad d_0 - \sigma \le r \le d_0 + \sigma$$
$$d = r - (d_0 - \sigma) \qquad r < d_0 - \sigma$$
$$d = r - (d_0 + \sigma) \qquad r < d_0 + \sigma$$

$r = |\boldsymbol{R}_i - \boldsymbol{R}_j|$, $\boldsymbol{R}_i$ and $\boldsymbol{R}_j$ are the positions of the atoms i and j that define the distance, d_0 is the desired position (arg6), k is the force constant (arg5) and σ is the half width of the flat-bottomed potential (arg7).

Note: With hydrogens bound to carbon in force fields MM2 and MM3, the restraint is applied between the van der Waals positions of the hydrogen (e.g., with a (C)H in MM2, the hydrogen is treated as if it were at a position shifted 8.5% of the C-H bond length toward the C).

If two `FXDI` commands specify the same atoms, the second replaces the first.

Note that for `MBAE` calculations, `FXDI` commands should be placed in a `.sbc` file.

arg1 *Atom 1*

0 Clear all existing `FXDI` constraints

arg2 *Atom 2*

arg4 *Nonbonded control*

1 The corresponding nonbonded interaction is eliminated from the interaction array. Used to allow fixed distances for atoms which are close in space.

arg5 *Force constant (default = 100 kJ/mol·Å^2)*

arg6 *Desired distance (Å)*

0 Use initial value.

>0 Use this value.

arg7 *Half-width of flat bottom part of the potential well*

FXBA — FiX Bond Angles

This command supplies energetic restraints to planar angles defined by the locations of triplets of atoms.

The constraint potential, E_c, for a fixed bond angle is calculated using the equation

$$E_c = k(\theta - \theta_0)^2/2$$

where θ is the current bond angle, θ_0 is the desired bond angle (arg6), and k is the force constant (arg5).

If two FXBA commands specify the same bond angle, the second replaces the first.

Note that for MBAE calculations, FXBA commands should be placed in a .sbc file.

arg1 *Atom 1*

0 Clear all existing FXBA constraints

arg2 *Atom 2*

arg3 *Atom 3*

arg5 *Force constant (default = 100 kJ/mol·rad^2)*

arg6 *Desired nonzero angle (degrees)*

0 Use initial value.

>0 Use this value.

arg7 *Half-width of flat bottom well (degrees).*

FXTA — FiX Torsion Angles

This command is used for restraining dihedral angles defined by the position of quartets of atoms. Such wells, particularly when the flat-bottomed option is used, are useful when minimizing crude structures in which some approximate torsional angle is known (e.g., from NMR coupling constants) but an exact value is not available. They are also useful in molecular dynamics to prevent major conformational changes without affecting the local conformational trajectory of the simulation. Using arg7=0., the same simple restraint in MacroModel to fix a torsion angle at arg6 is used.

If two FXTA commands specify the same torsion angle, the second replaces the first, unless arg8 is nonzero.

Note that for MBAE calculations, FXTA commands should be placed in a .sbc file.

arg1 *Atom 1*

0 Clear all existing FXTA constraints

arg2-4 *Atoms 2-4*

arg5 *V_1 Force constant (default = 1000 kJ/mol)*

arg6 *Desired torsion angle, in degrees*

>360 The initial value will be used.

arg7 *Half-width of flat bottom in degrees*

0 A standard 1-fold cosine well will be used.

arg8 *Multiplicity*

>1 Allows specification of several allowed ranges for the same set of torsion angles, using a method described by Sefler et al. [26]. If arg8 is two, for example, ranges around two dihedral angles can be specified in order to enforce constraints obtained from analysis of NMR coupling constants by means of the Karplus equation. In this example, two FXTA commands are expected for the same values of arg1-4. For each such FXTA command, you should specify a desired central position in arg6 and a flat-bottom half-width in arg7. When the last specification is read for a given atom set (for example, when the second FXTA interaction is read for an atom set with an arg8 value of two), MacroModel resets the central angles and half-widths for the set to values that achieve your original input specification. DEBG 14 makes this visible.

FXCO — FiX COnstraints

Generates a series of FXTA torsional constraints with flat bottoms to hold the input structure in its starting conformation. This opcode must come after any SUBS opcodes that are present and after the READ opcode that reads the structure for which these constraints are to be generated.

arg1 *Mode selection*

0 FXTA for all torsions.

1 FXTA for all but double bonds.

2 FXTA for all but bonds between sp^2 atoms.

arg2 *Clear prior constraints*

0 No (default)

1 Yes

arg5 *Force constant*

Default: 1000.

arg6 *Angular half-width*

Default: 60.

`FXHB` — FiX Hydrogen Bonds

Generates a series of distance (FXDI) and angle (FXBA) flat-bottomed constraints to constrain the geometry of hydrogen bonds.

arg1 *Donor Atom*

0 Constrain all hydrogen bonds. Possible hydrogen bonds are restricted to:
- Hydrogen - Acceptor distance < 2.5 A,
- D-H-A angle < 90 degrees,
- H-A-R angle < 60 degrees.

> 0 Donor atom number. If arg1 > 0, then:

arg2 *Hydrogen atom*

arg3 *Acceptor atom*

arg5 *H–A distance constraint value*

The half-width of the flat-bottomed constraint is set to 0.5 Å

0 use 1.85 Å

−1 use the current value

> 0 use this value

arg6 *D-H-A angle constraint value*

The half-width of the flat-bottomed constraint is set to 30°.

0 use 180°

–1 use the current value

> 0 use this value

Note: The H–A–R angle is not constrained.

4.8 Conformational Comparison

The commands in this section are used to describe criteria for considering conformations to be distinct. These commands are used in the contexts of conformational search and minimization of multiple conformations.

MSYM — use the MmSYM library

mmsym is a library that is used to identify a good map of atoms between two conformers for use in comparing these conformers. If the conformers are not dramatically different, this map will be the best one possible.

If mmsym is used, ATEQ, NSEQ, NSRO, and NSRF are unnecessary and are ignored. Comparisons can be done in place (molecules will not be translated or rotated while finding the map). If conformations are not being done in place, the structures written to the output structure file are positioned to minimize the RMS difference in the atomic positions of the comparison atoms between the first structure written and all subsequent structures.

Note: MSYM requires at least three atoms to be specified for comparison using the COMP operation code. The COMP opcode may appear before or after the MSYM operation code.

mmsym is the recommended mechanism for eliminating redundant conformers.

arg1 *Turn on/off mmsym*

–1 mmsym is not used. By default, mmsym is used.

arg2 *Turn on/off in place comparison*

1 In place comparison is used.

–1 In place comparison is not used.

By default, in place comparison is not used.

Relevant DEBG flag: 82

ADDC — ADD Conformation

ADDC is used to eliminate redundant conformers from a collection of conformers. Each time ADDC is executed it tries to add the current structure to the collection of conformers. The same checks for redundancy and transformations, such as net translation and rotation of the current structure, are performed as in a multiple minimization of conformers (see COMP) but no minimization or energy estimation is conducted. Energies from earlier MacroModel or Jaguar calculations may be used as part of the comparison process (see DEMX). If MacroModel energies are to be used then a FFLD line with the same force field as used to calculate the energies must precede the ADDC line.

arg1 *Energies to use*

–1 Use Jaguar energies from the input structure file.

By default, MacroModel energies are used if there is a FFLD line, otherwise energies are not used in the comparison process.

arg2 *Maximum number of structures to retain while running*

0 The maximum number of conformations to retain while running is 10,000.

>0 Specifies the maximum number of conformations to retain while running.

MULT — MULTiconformer minimization

Perform multiconformer minimization, i.e., delete duplicate structures as defined by the COMP command after minimization of each structure. If MULT or COMP is not used, then no elimination of duplicate structures will be done and every energy minimized structure will appear in the output file. If COMP is used without MULT, then multiconformer minimization is performed and the maximum number of structures saved is limited to 10000.

This command is appropriate only for a file of different conformations of the same molecule as would be produced by a previous conformational search. It most often is used with MINI to "polish" the results of a previous conformational search, using more stringent convergence criteria than were used in the search itself.

Note: MULT must appear before MINI in the .com file.

arg1 *Maximum number of conformers that can be saved*

0 10000 (default)

n>0 *n*

CHIG — CHIrality checking (Global)

This command identifies and saves the chirality of the atoms listed in the arguments, and rejects structures whose chirality has changed during the calculation. The chirality is stored only for the first structure in the input file. All structures are subsequently checked against the chirality of the first. Used with MULT, MCSM, MCMM, LMCS, LMC2, and LOOP for files of identical structures differing only in conformation.

Use as many CHIG commands as necessary to save all important chiralities.

CHIG commands must come after the READ command in the .com file. With MCMM searching, chirality is checked after the Monte Carlo step and also after the following energy minimization.

arg1 *Atom number of chiral center*

0 Automatically perceive chirality. See the AUTO opcode. arg2-4 are ignored.

>0 Atom number.

arg2-4 *Atom numbers of chiral centers*

>0 Atom number.

arg8 *Chirality reporting*

1 Report chirality identification.

COMP — structure atom COMParison

Arg1-4 are atom numbers to be used for comparing a minimized structure with all previous unique minima found. COMP may be specified up to 50 times to allow up to 200 atoms to be used in the comparisons. If COMP is not specified, the program will not attempt to eliminate duplicate minima. Structures are considered the same unless the least squares superimposition of the compared atoms finds one or more pairs of equivalent atoms separated by more than the separation given by the CRMS command (default = 0.25 Å). It is not necessary to specify all atoms in a molecule for comparison, but a representative sampling from widely separated points in the structure should be given in COMP commands.

arg1 *First atom number for comparison*

0 Generate comparison atom list automatically based on arg7 values.

arg2-4 *Additional atom numbers for comparison*

arg7 *Comparison list inclusion*

0 Include heavy atoms.

1 Include heavy atoms and hydrogens bonded to oxygen atoms.

2 Include all atoms.

3 Include only ring atoms. If no ring atoms are found, include heavy atoms and hydrogens bonded to oxygen atoms.

arg8 *Comparison atom reporting*

1 Report comparison atom identification.

`CRMS` — Convergence RMS

Sets the geometric criterion which defines two structures to be identical within a conformational search or a multiple minimization. Despite the command name, the program uses as its criterion not the root-mean-square interatomic distance after optimal rigid-body superposition of a pair of structures, but rather the maximum distance between corresponding atoms after superposition.

arg1 *Maximum atomic separation (Å)*

If very small deviations are used (e.g., arg1 = 0 or 1), then some duplicate structures may be retained due to incomplete convergence in energy minimization.

0 .25 (default value)

1 .10

2 .20

n 0.1*n*

arg5 *Maximum energy difference for geometric comparison*

If two structures differ by an energy greater than this value, geometric comparison is not attempted; the structures are considered different based on the fact that the energies differ.

0 4.184 kJ/mol.

<0 0 (All structure pairs will be compared geometrically, regardless of the energy difference between them.)

arg6 *Distance selection criteria for conformational structures*

A floating point value can be placed into arg6. This overrides any arg1 setting.

>0 This value, in angstroms, is the maximum distance apart any two structures can be to be considered the same during conformational comparison. As before, despite the name of the command, the criterion used is the maximum (not RMS) distance between corresponding atoms following optimal rigid-body superposition.

<0 If arg6 is given a negative value, no conformational comparisons are done, and all conformational pairs are considered dissimilar

arg7 *Maximum difference in dihedral angle*

If AUTO arg2 is set to 4 then the dihedral angles involving polar hydrogen atoms (those bonded to a O, S or N atom) are used in determining whether conformers are redundant or not. If the dihedral angles differ by more than the value specified by this argument then the conformers are judged to be distinct. Sets polar H dihedral angle the (default: 60˚)

0 use default (60˚)

> 0 tolerance in degrees (must be less than 180˚).

arg8 *Comparison criterion*

0 Use maximum distance criterion.

2 Use RMSD for comparison rather than maximum distance.

ATEQ — ATom EQuivalencies

This command is used with COMP commands to allow the identification of nonunique structures having symmetrical atoms (e.g., the two oxygens of a carboxylate ion or the ortho/meta pairs of carbons of a phenyl ring). We suggest using MSYM rather than specifying atom equivalences using ATEQ. If such equivalent atoms are present (and listed as comparison atoms in COMP commands), then identical conformers having different atom numbering systems will emerge as different, unique final structures. To avoid this duplication, ATEQ commands are used to note equivalent atoms. A different ATEQ command is used for each equivalent atom set and may equivalence up to 4 atoms in each set.

ATEQ commands normally only cyclically permute the atoms listed for comparisons of conformations. However, search techniques such as LMOD (LMCS) and LLMOD (LMC2) can result in non-cyclic permutations. Specifying DEBG 79 causes non-cyclic permutations to also be considered in comparisons of conformations.

For example, if a molecule has four carboxylates and a trimethyl ammonium, then one would include five ATEQ commands, 4 for the carboxylates (2 equivalent atoms) and 1 for the trimethyl ammonium (3 equivalent atoms) if all atoms were included in COMP commands. Alternatively, such symmetrical atoms may be left out of the comparison lists. For high symmetry cases, it is better to replace ATEQ with NSEQ, NSRO, and NSRF commands. In such cases a better alternative would be to use MSYM.

In applying the ATEQ commands, the program will generate all permutations of equivalenced atoms to try for a near perfect geometrical match (i.e., find a duplicate conformer). Such an approach will generate, inter alia, some nonsense permutations with clustered equivalent atoms (e.g., phenyl rings) but such permutations will never match and do not cause problems.

Turning on DEBG 81 causes full information to be printed.

arg1 *Atom number of atom having other equivalent atoms*

arg2-4 *Atom numbers of atoms equivalent to arg1*

NSEQ — Numbering System EQuivalencies

We suggest using MSYM rather than specifying atom equivalences using NSEQ.

This command allows the user to list alternative numbering systems for the molecule. For each alternative numbering system, you will need as many NSEQ commands as you have COMP commands. The COMP command can be considered as the original numbering system of the comparison atoms. Each block of NSEQ commands corresponds an alternative numbering system for the comparison atoms listed in the COMP command. For united atom butane for example, the COMP command might contain arg1-4 as 1 2 3 4, then the NSEQ command would contain as the only possible alternative numbering system 4 3 2 1. An alternative to this simple case would be to use NSRF. See the *MacroModel User Manual* for more complex examples.

arg1-4 *Equivalent atom numbers*

NSRO — Numbering System ROtation

We suggest using MSYM rather than specifying atom equivalences using NSRO.

This is intended primarily for use on symmetrical cyclics. This feature will cause the program to compare not only corresponding atoms, but also all possible rotations of the numbering system. Used with completely symmetrical systems such as cycloalkanes with the comparison command, arg1 should be 1. Used with systems like 18-crown-6, arg1 should be 3 (note that this will not eliminate all duplicates with 18-crown-6—to do this, use the NSEQ commands). NSRO automatically turns on the NSRF option so that enantiomeric conformations are eliminated (this feature can be disabled with the NANT command). All atoms in the ring being

rotated must be listed in the COMP commands. Furthermore the atoms listed in COMP commands must be in the order in which they occur in the ring. Atoms not in the ring (hydrogens or other substituents) should not be listed.

NSRO must come before READ or TRED commands and after COMP commands.

arg1 *Rotation increment*

The number of atoms by which a ring must be rotated to bring equivalent atoms into superimposition. (Default: 1).

NSRF — Numbering System ReFlection

We suggest using MSYM rather than specifying atom equivalences using NSRF.

If invoked, this feature will cause the program to compare not only corresponding atoms but also numbering system reflections. Used with symmetrical structures such as normal acyclic alkanes with the comparison command. This command reverses the ordering of the atoms in the COMP commands for comparison purposes, thus atoms in the COMP commands must be given in the order of the atoms in the chain.

NSRF must come before READ or TRED commands.

NANT — do not consider eNANTiomers to be duplicates

Ordinarily, enantiomers are considered identical for the purpose of conformational comparisons. This command, which alters this behavior, takes no arguments.

DEMX — Delta-E MaX energy windowing

DEMX sets a window for permissible energy above the lowest-energy conformation. This command discards any structure that is more than arg5 kJ/mol above the minimum energy conformation. Since more than one force field may be used in a command procedure, arg1 is a code that is matched to a corresponding "energy code" in a MINI command. DEMX is used with MULT and MCMM commands. If no DEMX is used, then all energy minima will be kept regardless of their relative energies. We use this command to limit the energy range of the conformers printed out at the end of the procedure. We suggest a value of 25.0 kJ/mol (*ca.* 6 kcal/mol) for arg5 to prevent output of high-energy structures. If you plan to do a subsequent solvation treatment, a 50.0 kJ/mol window may be more appropriate to allow for major reordering of structures on inclusion of solvation. However, it is better to include solvation (SOLV command) directly in the minimization.

A second energy window may be set in arg6 (suggested value 1.5-2 times that in arg5), which is used to reject structures before complete minimization by a check of the relative energy at iteration number arg2. Aborting the minimization of structures whose energies appear too large part way through the minimization makes the overall procedure faster.

arg1 *Identifier*

This is an arbitrary integer that must have the same value as arg4 in some MINI command. In most typical use, both are zero.

arg2 *Number of iterations before preliminary energy test*

After this number of minimization iterations are done, a test will be performed to see whether the current structure is less than the arg6 kJ/mol above the global minimum found so far. A conservative value is 1/3-1/2 the number of iterations in the MINI command (arg3). Default: a very large number, implying that no preliminary test will be performed.

A reasonable value for arg2 is 1/3-1/2 of the number of iterations to give minimization convergence for a typical conformation, but it is wise to minimize several conformations of the actual structure to see how many iterations are necessary to bring the energy to within several kJ/mol of its final value.

arg5 *Energy window for final test*

Default: 0.0 (i.e., only the global minimum will be kept).

arg6 *Energy window for preliminary test*

A conservative value is twice arg5. Default: 0.0 (i.e., all minima will be kept).

NORE — NO REordering done on output structures

Normally, MULT and conformational search output structures are reordered in increasing energy. If this command is specified, this reordering is suppressed.

NORE must come before READ or TRED commands.

4.9 Coordinate Manipulation

The combination of the COPY and ALGN opcodes permits very rapid approximate alignment of ligands with a reference ligand using center of mass and moments of inertia. These commands may be used to crudely preposition ligands prior to an Embrace conformational search calculation based upon the position of a reference ligand already positioned in the active site. While this combination of commands may be useful the positioning is crude and the searching of conformational space is slow and quite limited compared to that available in Schrödinger's docking program, Glide.

Note that COPY/ALGN may require the purchase of a separate license.

COPY — COPY coordinates

Copies the current set of coordinates to a reference array for later use. The reference array is currently only used by the ALGN command.

ALGN — ALiGN with reference coordinates

Aligns the current system's coordinates with the reference system coordinates. Alignment can be conducted by center of mass or principal axis or both. The reference coordinates must have already been saved using COPY. ALGN may be used in a BGIN/END loop and is intended to pre-position molecules prior to further processing with commands like MINI and MBAE. To save aligned structures in the output structure an explicit WRIT statement is needed unless arg3 is 5 (see below).

If you use ALGN with MBAE and want to specify substructures with SUBS, you must set arg2 of SUBS to 1 to ensure that the correct substructures are used.

arg1 *Type of alignment*

0,1 Reposition the current system so that its center corresponds to that in the reference system.

2 Reorient the current system so that the principal axes of the current system correspond to those in the reference system.

3 Reposition and reorient the current system so that its center and principal axes correspond to those in the reference system.

arg2 *Atom weighting*

0, 1 Mass-weight the coordinates when calculating the system center and principal axis.

2 Weight all atoms equally when calculating the system center and principal axes.

arg3 *Principal axis alignment*

This argument is only used if arg1 is 2 or 3.

0, 1 Closest principal axis alignment.

2 Rotate by 180 degrees about the first principal axis relative to the closest principal axis alignment.

3 Rotate by 180 degrees about the second principal axis relative to the closest principal axis alignment.

4 Rotate by 180 degrees about the first principal axis and then rotate by 180 degrees about the second principal axis.

5 Generate all 4 alignments with respect to the principal axis and write each alignment to the output structure file.

4.10 Conformational Searching

The entire contents of the input file are used as the "seed" for the search for most search methods. At the start of a run, the full contents of this file is read, and the search proceeds as if all the structures in the input file had been found in the current search. This allows a new search to use the results of a previously run, partial search as its input. Of course, for a new search, the input file will contain a single structure.

The output file from the search contains in the header lines of the individual structures the number of times each structure was found. Therefore, the full benefits of usage-directedness are obtained in a subsequently run search. This degeneracy information is ignored, for technical reasons, during a network-distributed search. In a distributed search, or in any search seeded from an output file produced by an earlier version of MacroModel, the structures are treated as if they each occurred exactly once in a previous search. Debug (`DEBG`) flag 360, if set, instructs the program to ignore the degeneracy information even if it would otherwise utilize it.

Searches may be updated during program execution. When a search is updated, the information in the temporary file (*filename*`.tmp`) is written to the output file, after discarding structures that are not within the specified energetic window of the current global minimum. A summary of the job progress is also printed, which includes information on the number of times various structures were found and the convergence of these structures during minimization. As described in Chapter 2, updates can be initiated interactively by the user. Automatic, periodic updates can be controlled with the `MCOP` command.

For a discussion of conformational search protocols, see the Chapter 9 of the *MacroModel User Manual*. As discussed there, it is advisable to perform the search without exhaustive minimization of all found structures and to subsequently reminimize the surviving structures.

This can be done without initiating a new MacroModel job, by inserting into the `.com` file a `RWND` command following the `MINI` command that terminates the search setup, then inserting a `BGIN/END` loop within which a `READ/MINI` sequence is specified. The `COMP` atoms used in the search will continue to be active during the reminimization.

The maximum number of conformations that can be stored when using `MCMM`, `LMCS`, `SPMC`, or `MULT` is no longer limited to 10000. See the respective command descriptions and Section 1.4 on page 9 for further details.

AUTO — AUTOmatic setup

Perform automatic setup for MCMM, MINI, SPMC, LMCS, LMC2, and combinations of MCMM with LMCS or LMC2, on the current structure. This procedure takes into account the current substructure information. The setup for MCMM is compatible with MCMM/LMCS and MCMM/LMC2. In addition, it can be applied in a serial manner (across different molecules in the input structure file) in MCMM, MCMM/LMCS, and MCMM/LMC2 serial searches, but not in pure LMC2 serial searches.

AUTO is currently incompatible with LOOP and must be used carefully with MBAE. In addition, AUTO is incompatible with frozen atoms unless a substructure is being used.

Note: Serial searches can generate large numbers of output structures, and the resulting output structure file can be large enough to exceed file size limits.

AUTO should be preceded by the MSYM, MCMM, MCNV, MCOP, LMCS, LMC2, SPMC, SUBS, and READ opcodes, if present, and is best placed just before the MINI line in the .com file.

The type of calculation in use is detected automatically. For MINI calculations AUTO sets up comparison atoms, chiral atoms, and torsional constraints unless instructed not to. For MCMM calculations AUTO sets up comparison atoms, chiral atoms, torsional constraints, molecule moves, variable torsions, and ring closures. For LMCS and LMC2 calculations, molecule moves, variable torsions, and ring closures are set up but not used, unless the setup is turned off. Series of bonded atoms that connect fixed or frozen atoms in substructures have ring closure bonds created within them to prevent rotation of fixed or frozen regions. Failure to find such a ring closure turns off all variable torsions in the SUBS atoms connecting such regions.

AUTO arg8 controls the extent to which variable torsions are applied to the input structures. This option is relavent mostly to amide and ester linkages, and supplies a greater degree of control over sampling of standard and non-standard amides and esters. arg8=1 applies torsion moves to the amide/ester linkage of non-standard groups like anhydrides, carbamates, hydrazones, and so on. Normal esters and amides are not sampled with this setting, but a torsional constraint is applied to them to ensure that the relative conformation of these groups is maintained. arg8=2 adds torsion moves to sample normal esters and amides in addition to amide and ester linkages of non-standard groups. arg8=3 adds torsion moves to C=N and N=N bonds, in addition to all amide and ester derivatives.

In automatic setup, the maximum number of torsions varied at the same time is set to the minimum of arg2 of MCNV and the number of torsions present in the current system. If MCNV arg2 has not been set then the number of torsions present in the current system is used. In serial calculations, the number of torsions varied at the same time is determined for each system.

arg1 *AUTO On/Off*

−1 Turn off automatic setup.

0 Turn on automatic setup and generate a new setup using the current system.

1 Turn on automatic setup. If automatic setup is already on, use the existing setup.

arg2 *Comparison atom setup*

−1 Use existing list of comparison atoms.

0 Use previous setting. If none exists, arg2 is set to 2.

1 Create a new list of comparison atoms containing all non-hydrogen atoms.

2 Create a new list of comparison atoms containing all non-hydrogen atoms and hydrogen atoms bound to oxygen atoms.

3 Create a new list of comparison atoms containing all of the atoms.

4 Create a new list of comparison atoms containing all non-hydrogen atoms. Also use dihedral angle differences involving polar H atoms (bonded to a O, S or N atom) in redundant conformer elimination. If the dihedral angles in two conformers differ by more than a threshold (controlled by `CRMS` arg7) then the conformers are judged to be distinct.

5 Create a new list of comparison atoms containing all ring atoms. If no ring atoms are found, use heavy atoms and hydrogen atoms bonded to oxygen atoms.

arg3 *Chiral atom setup*

−1 Use existing list of chiral atoms.

0 Use previous setting. If none exists, arg3 is set to 1.

1 Create a new list of chiral atoms (overrides `CHIG` for the current structure).

arg4 *Torsional constraints*

−1 Use existing list of torsional constraints.

0 Use previous setting. If none exists, arg4 is set to 1.

1 Create a new list of torsional constraints for carbon-carbon double bonds, amides, and esters.

2 Create a new list of torsional constraints for carbon-carbon double bonds only.

arg5 *Torsional selection*

–1 Use existing list of torsions.

0 Use previous setting. If none exists arg5 is set to 1.

1 Create a new list of torsions to sample.

arg6 *Serial calculation*

–1 Turn off serial mode for MCMM calculations.

0 Use previous setting.

1 Turn on serial mode for MCMM calculations. Serial MCMM calculations are off by default. If LMCS serial calculations have already been requested, then serial MCMM calculations are also turned on automatically when the AUTO opcode is used regardless of the value of arg6.

arg7 *Minimum ring size for ring closures and torsional sampling*

0 Use the previous setting. If there is no previous setting, use the default setting.

>3 Attempt to sample rings with at least this many atoms as members. The default value is 5 (five-membered rings or larger).

arg8 *Sampling of torsions around bonds involving planar groups*

Specify sampling of torsions around amide and ester linkages, azo (N=N) and imino (C=N) groups.

0 Do not sample torsions around planar groups.

1 Sample torsions around non-standard amide and ester linkages; do not sample other planar groups

2 Sample all amide and ester linkages; do not sample azo and imino linkages.

3 Sample all amide and ester linkages, and sample azo and imino linkages.

Related DEBG flags: 520, 521.

AUOP — AUto OPtions

When AUTO is used in conjunction with an MCMM, mixed MCMM/LMCS, mixed MCMM/LMC2, SPMC, or CGEN conformational search, AUOP may be used to modify the number of steps depending on some measure of the number of degrees of freedom present. AUOP should be used immediately before the AUTO command.

For CGEN by default the number of moves can also depend on ConfGen's internal estimate for the number of degrees of freedom. See CGO2 arg3 for more information.

arg1 *Number of times global minimum is found*

>0 The search will terminate when the global minimum is found this many times. Not recommended for ConfGen searches.

arg3 *Exclude variable chirality atoms from chiral atom list*

0,1 Use full list of chiral atoms.

2 Exclude atoms whose chirality can vary in solution. This currently only excludes nitrogen atoms bonded to 4 atoms one of which much be a hydrogen atom. Not recommended for ConfGen searches.

arg5 *Number of steps*

<1 The search will use the number of steps specified by the search command itself, e.g. arg1 of MCMM .

>1 The search will use (number of degrees of freedom) * arg5 steps if this is less than the number of steps specified by the search command itself. For MCMM, mixed MCMM/LMCS, mixed MCMM/LMC2 and SPMC searches the number of degrees of freedom is given by the number of torsions (TORS) and molecules moved (MOLS) identified by AUTO. For CGEN jobs, the number of degrees of freedom can be estimated this way or from ConfGen's internal estimate for the number of degrees of freedom, depending on the value of CGO2 arg3.

LMCS — Low-Mode Conformational Search

This method is described by Kolossváry and Guida [27, 28]. In tests so far, it has proved highly efficient, and it has the advantage that ring structures and variable torsion angles do not have to be specified. LMCS works by exploring the low-frequency eigenvectors of the system, which are expected to follow "soft" degrees of freedom, such as torsions. It is, however, helpful to specify independently movable molecules, by means of MOLS commands. Like the MCMM procedure, LMCS may be seeded or restarted with multiple input structures, and LMCS may also

be used with distributed MacroModel. Most features that work with MCMM also work with LMCS. COMP, CHIG, DEMX, and MCSS commands should ordinarily be specified, and several MCOP arguments take on special meaning when used with LMCS.

The AUTO opcode can be used for automatic setup of LMCS and MCMM/LMCS calculations.

LMCS comes in two varieties: LMCS-SHAKE and mode-following. LMCS-SHAKE follows the low-mode vector, but every 2 Å it applies a few steps of steepest descent minimization to relieve any strain due to distorted bond lengths and bond angles introduced by the move.

The mode-following (eigenvector-following) option allows the user to apply the original low-mode search concept: "Since the potential energy hypersurface is a network of interconnected minima and saddle points, we reasoned that one could utilize a procedure that relies on eigenvector following for conformational searching. Thus, one could initiate the search by starting with any local minimum. By using one of the eigenvector-following techniques, one could locate a saddle point associated with this minimum and then the other minimum associated with this saddle point. By application of the eigenvector-following technique to the second minimum or to a different eigenvector of the first minimum, additional minima could be located which could then be used to find additional saddles, etc." (quoted from the *J. Comp. Chem.* article cited above).

Mode-following uses the same eigenvector-following saddle point search method as in the SDLP command (see details there). If LMCS mode-following locates a saddle point, it proceeds as follows. The eigenvector for the negative eigenvalue is computed, followed by a short move along that eigenvector away from the starting structure. The resulting structure, which is slightly lower in energy than the saddle point structure, is then energy-minimized to locate the minimum energy structure on the other side of the saddle point.

If LMCS mode-following cannot locate a saddle point, the starting (minimum) structure is restired and is automatically "rejected by starting geometry." Thus, only minima found via saddle points are stored. This makes LMCS mode-following useful for the mapping of conformational interconversions in a local region of the potential energy surface. The recommended procedure for this is to set LMCS arg5=–1 (every step mode-following), and MCSS arg1=0 (random walk structure selection) or LMCS arg4=1 (local search mode).

Note: LMCS mode-following is not intended for general conformational searches. Simple LMCS or LMCS-SHAKE (LMCS arg5>=0) is much more efficient for this purpose. LMCS mode-following is only recommended for local searches exploring the conformational interconversions of a molecule. DEBG 920 saves all the intermediate structures during LMCS-SHAKE or mode-following in the output structure file and colors them so that you can conveniently visualize the different multiple LMCS moves.

LMCS can be combined with MCMM search. Tests have confirmed that the most efficient use of LMCS is to allow explicit torsional rotation of key torsion bonds, especially in acyclic structures. Therefore, you can combine LMCS with TORS commands (with RCA4 if necessary) without restriction. MOLS commands can also be applied to translate/rotate independently movable molecules. See also the MCOP command.

arg1 *Number of Monte Carlo steps to be carried out before stopping*

0 Carry out the search until arg2 structures have been found. We do not recommend the use of this default.

arg2 *Maximum number of structures to retain while running*

0 The maximum number of conformations to retain while running is the value specified in arg1 or 10,000, whichever is greater.

>0 The maximum number of conformations to retain while running.

arg3 *Number of low-frequency modes*

LMCS will explore the first |arg3| number of modes.

0 Default: 10 (pure modes).

<0 LMCS explores a random linear combination of the first |arg3| number of modes, rather than exploring single pure modes at a time.

arg4 *Search mode*

0 Global search (each Monte Carlo step begins with the preceding Monte Carlo structure, providing the structure is within 100 kJ of the global minimum). This is equivalent to MCSS (arg1=0, arg2=0, arg4=100.0).

1 Local search (each Monte Carlo step begins with the original structure). Generally used with small coordinate variations in TORS or MOLS to find other minima which are closely related to the starting structure.

arg5 *Control of multiple LMCS steps*

0 Default: Single LMCS leap (MacroModel 6.0 behavior).

>0 Frequency of using LMCS-SHAKE (must be in range 0 to 1).

<0 |arg5| is frequency of using mode-following (arg5 must be in range 0 to −1).

arg6 *Allowable interatomic approach distance*

Fraction of sum of van der Waals radii used as a closest atomic approach limit. If a van der Waals pair comes closer together than this as the result of an LMCS move, the move is rejected before minimization.

0 Default: 0.25

>0 Other fraction.

arg7 *Minimum distance (Å)*

In an LMCS move, a random total travelling distance is selected between the specified minimum and maximum values. The distances specified here and in arg6 correspond to the motion of the fastest-moving atom.

It is often useful to perform a short conformational search with the default values of arg7 and arg8, and, based on the results, adjust these arguments accordingly. If many conformations minimize back to the starting conformation, increase arg7 (and perhaps arg8). If many conformations are ruled out because of distorted sp^3 carbons, decrease arg8 (and perhaps arg7). The information needed to make these choices will become visible by specifying `MCOP` arg1=1.

0 Default: 3 Å

>0 Other value to be used.

arg8 *Maximum distance (Å)*

0 Default: 6 Å

>0 Other value to be used.

`LMC2` — Low-Mode Conformational search for large molecules

This method, termed `LLMOD`, is described by Kolossváry and Keseru [29]. `LLMOD` has been developed for large-scale conformational searching, such as protein loop optimization, homology model refinement, and fully flexible docking for induced fit modeling, only to mention a few applications [30]. `LLMOD` is very similar to `LMOD`, which is implemented in MacroModel as the `LMCS` command, but `LLMOD` utilizes the ARPACK package to compute low-mode eigenvectors of a Hessian matrix that is only referenced implicitly, through its product with a series of vectors. The Hessian × vector product can be calculated by a number of methods. `LLMOD` is the first conformational search method that can be applied to fully flexible, unconstrained protein structures.

LMC2 is implemented to extend the basic use of LMCS to molecules of virtually any size. Note that two main features of LMCS, LMCS-SHAKE and LMCS mode-following, are not implemented in LMC2. The command arguments of LMC2 are fairly similar to those of LMCS. Like LMCS, LMC2 can also be used in conjunction with the MCMM command to explicitly alter key torsion bonds via TORS commands (with RCA4 to open macrocyclic structures if necessary) and apply explicit translation/rotation of a ligand with the MOLS command for docking. LMC2 is primarily designed for fully flexible molecules, but the use of frozen atoms is allowed, and recommended in cases where parts of a macromolecule can be treated rigidly. Also note that you can set certain parameters controlling the ARPACK package via the ARPK command.

The AUTO opcode can be used for automatic setup of LMC2 and MCMM/LMC2 calculations. It can be used for MCMM/LMC2 serial searches but not for pure LMC2 serial searches.

LMCS is more efficient than LMC2 when the number of atoms is small enough for the entire Hessian to be stored in computer memory without swapping.

Note: It has proven to be good practice and time-saving to minimize molecular structures during a conformational search to a gradient that is not too low and only re-minimize entirely the final set of low-energy conformations. In that respect, LMC2 needs special attention. It is still recommended to follow this scheme, e.g., for unconstrained proteins, a gradient RMS of 1 kJ/mol/Å is sufficient during the search, but the very first structure, for which the low-modes are computed, must be pre-minimized to a low, < 0.1 kJ/mol/Å gradient RMS in order to derive reasonable modes. Pre-minimization can either be done in a separate MacroModel job, or the RWND command can be used to fully minimize the first structure, rewind the output file and start the LMC2 conformational search from there.

Note: Large-scale low-mode calculations can use a large amount of memory. If memory issues are encountered, consider eliminating the use of solvation. In addition, reducing non-bonded cutoffs also reduces the memory required for the computation.

arg1 *Number of Monte Carlo steps to be carried out before stopping*

0 Carry out the search until arg2 structures have been found. We do not recommend the use of this default.

arg2 *Maximum number of structures to retain while running*

0 The maximum number of conformations to retain while running is the value specified in arg1 or 10,000, whichever is greater.

>0 This is the maximum number of conformations to retain while running.

arg3 *Number of low-frequency modes*

LMC2 explores the first |arg3| modes. A maximum of 100 modes is allowed.

0 Default: 30 (pure modes).

<0 LMC2 explores a linear combination of all |arg3| low-frequency modes, rather than exploring single pure modes at a time. Note that LMC2 employs a better mixing strategy than LMCS, which simply uses a random mixture of low modes. LMC2 perturbs the starting structure by moving along the first |arg3| modes successively. Each move is carried out exactly the same way as for a selected pure mode, but each move employs a different random distance (see arg7/arg8).

arg4 *Search mode*

0 Global search: each Monte Carlo step begins with the preceding Monte Carlo structure, providing the structure is within 100 kJ of the global minimum. This is equivalent to MCSS (arg1=0, arg2=0, arg4=100.0).

1 Local search: each Monte Carlo step begins with the original structure. Generally used with small coordinate variations in TORS or MOLS to find other minima which are closely related to the starting structure.

arg5 *Determines when low-mode eigenvectors are calculated*

ARPACK calculations on proteins are time consuming. The time span of a single low-mode calculation can vary from a few minutes to an overnight job. It is highly recommended that eigenvectors be re-calculated only when it is absolutely necessary, i.e., when there is evidence that low modes calculated for a particular conformation cannot be transferred to other conformations. In other words, as long as a single set of low modes can produce efficient search directions starting from a number of different local minima, there is no need to calculate a different set of modes for every new conformation.

0 Default: Calculate the low modes for the very first minimized structure only and use those modes throughout the entire search (recommended).

>0 Low-mode eigenvectors are recalculated every time the search finds a new global minimum. The modes of the current global minimum are always used.

<0 Low modes are recalculated for every new structure (LMCS behavior).

arg6 *Allowable interatomic approach distance*

Fraction of sum of van der Waals radii used as a closest atomic approach limit. If a van der Waals pair comes closer together than this as the result of an LMCS move, the move is rejected before minimization.

0 Default: 0.25

>0 Other fraction.

arg7 *Minimum distance (Å)*

Same as LMCS, except that the default values are different, depending on arg3.

0 Default: 3 (Å) for single mode, arg3 >= 0

1 (Å) for mixed mode, arg3 < 0

>0 Other value to be used.

arg8 *Maximum distance (Å)*

Same as LMCS, except that the default values are different, depending on arg3.

0 Default: 6 (Å) for single mode, arg3 >= 0

2 (Å) for mixed mode, arg3 < 0

>0 Other value to be used.

LOOP — protein LOOP conformation generation

LOOP is a conformational search method for protein loops that works in conjunction with the MINI opcode in a manner similar to MCMM.

Given a single protein structure LOOP attempts to generate candidate loop conformations using either the specified loop from the protein structure provided or an alternate user specified sequence for the loop. LOOP rapidly generates a candidate structure with the appropriate bond lengths and angles. In addition, heavy atoms in this structure (i.e., atoms heavier than hydrogen) will not approach each other more closely than a specified distance. These candidate structures are then energy minimized and can be checked for uniqueness in a manner analogous to that for MCMM. A more general discussion of the use of LOOP is available in Chapter 13 of the *MacroModel User Manual*.

`LOOP` generates candidate structures in two stages:

- Initial loop generation via random selection of rotatable dihedral angles without concern for loop closure
- "Tweak," an iterative process involving the systematic adjustment of the rotatable bonds to attain loop closure [31, 32] while eliminating very close encounters amongst heavy atoms.

In the loop generation stage, the N-terminus and C-terminus halves of the loop are constructed separately, starting from their attachment points, by growing the chain out to the next rotatable bond. Then a random dihedral angle is chosen for this bond and the heavy atoms positioned unambiguously as a result are checked for close approaches with heavy atoms in the rest of the protein (not the loop itself). If there are such close approaches, the process of selecting a new random angle and checking for close approaches is repeated until either N_{try} such attempts are made or no close approaches result. If N_{try} such attempts fail then the loop construction process retreats back to the previous rotatable bond for which a new random angle is chosen. When a dihedral angle has been chosen successfully, the process focuses on the next rotatable bond out from the attachment point. Rotatable bonds in side chains are also sampled in this manner. The loop generation stage produces a random structure for the protein loop that is not connected in the middle and does not have close approaches between the loop atoms and the rest of the protein, but there may such close approaches within the loop itself.

In the second stage the Tweak algorithm is used to bring the middle of the loop together in the right geometry. Briefly, this algorithm proceeds by iteratively selecting a set of minimal changes to the rotatable bonds that lead toward loop closure based upon a linear approximation. Loop closure is specified by a set of target distances that must be met to within a tolerance of 0.001 Angstroms. Along the way close approaches involving heavy atoms of the loop atoms and the rest of the protein are removed by temporarily introducing additional target distances between the overlapping atoms. When the set of distances for closing the loop are met to within 1 Angstrom the Tweak algorithm also attempts to remove close approaches amongst the heavy atoms within the loop. By using a minimal set of changes to the rotatable bonds the Tweak algorithm retains as much of the original, random loop conformation in torsional space as possible and thus provides a mechanism for generating a diverse set of candidate loop structures.

Limitations:

- Only one loop can be treated, although successive calculations may target different loops.
- An all-atom representation of the protein must be used.
- The protein loop and the residues that it is immediately attached to must consist of alpha amino acids.

- Rings (e.g., the one in proline) are treated as rigid.
- No disulfide bonds are permitted within the loop or between the loop and the rest of the protein.
- No atoms in the loop can be frozen.

Using LOOP usually results in a renumbering of the atoms within the structure such that the loop atoms become the highest numbered atoms in the structure. LOOP tries to make this work by automatically shifting the atom numbers provided in SUBS, FXAT, FXDI, FXBA, FXTA, COMP, and CHIG commands. There are also routines for automatically generating COMP and CHIG atoms for the loop atoms as described for arg3, below. To facilitate subsequent studies using the structures generated, a new substructure file, *out_filename*.sbc, containing the shifted atom numbers is generated automatically. In addition, shifted COMP and CHIG commands suitable for use in .com files are written in the .log file.

If comparison atoms (COMP) are specified explicitly or implicitly (see arg 3) then MSYM must be used.

arg1 *N-terminus atom number for the loop*

This is the atom number for the peptidic Nitrogen atom that joins the loop to the protein at the N-terminus of the loop. This must be nonzero.

arg2 *C-terminus atom number of the loop*

This is the atom number for the peptidic Carbon atom that joins the loop to the protein at the C-terminus of the loop. This must be nonzero.

arg3 *Type of LOOP Generation*

–1 Turn off LOOP and clear out information specified in an earlier loop command. All other arguments are ignored.

0 Generates alternate conformations for the loop specified by args 1 and 2. In this mode the first structure minimized is that originally provided.

1 Remove the loop specified by args 1 and 2 and replace it with the amino acid sequence specified in an *in_filename*.lsq file. Candidate loop structures will be generated for this new loop. This file consists of a list containing codes for the amino acids; one to a line starting from the N terminus of the loop. For the 20 common amino acids the standard 3-letter codes are used. However, the code for any amino acid present in Maestro's modifiable fragment tables may be given provided that it is an alpha amino acid.

arg4 *The number of loop conformations to generate*

0 Default: 100

arg5 *The maximum number of loop conformations to keep*

0 Default: 10000

arg6 *Allowable interaction approach distance*

If a given pair of atoms is closer than arg6 times the sum of their van der Waals radii after loop generation just prior to minimization then the structure is rejected and another loop structure is generated.

0 Default: 0.25

arg7 *Automatic generation of COMP and CHIG atom lists for the loop*

–1 Don't generate COMP or CHIG commands for the atoms in the loop.

0 Automatically generate COMP commands for the heavy atoms in the loop and CHIG commands for atoms in the loop.

1 Automatically generate COMP commands and CHIG commands for all the atoms in the loop.

2 Automatically generate COMP commands for the heavy atoms in the loop and do not generate any CHIG commands.

3 Automatically generate COMP commands for all the atoms and do not generate any CHIG commands.

Related DEBG flags: 555

LPOP — LooP OPtions

LPOP provides a means to modify some of the parameters related to protein loop generation with the LOOP opcode. If no LPOP commands are provided the default values for the parameters are used. Multiple LPOP commands (e.g., with different arg1 values) may be used. It is not expected that these parameters would need to be modified from their default values under normal conditions so typically one would not need to include a LPOP command in the .com file.

arg1 *Parameter set modified by LPOP*

This argument controls which set of parameters is modified by the remaining arguments to `LPOP`.

0 Tweak parameters

arg 2 *Maximum number of Tweak iterations that can be used while trying to generate an acceptable loop conformation.*

Default: 600

arg5 *Early check iteration fraction.*

Loop generation attempts will be abandoned at arg5 * arg2 tweak iterations if the distance criteria specified by arg6 is not met.

Default: 0.167

arg6 *Early check maximum deviation.*

If the maximum deviation of the distances from their desired values is greater than the value of arg6 the current tweak process is abandoned and the process for generating a new loop is restarted.

Default: 7.0 Å

arg7 *Maximum ratio of constraints to rotatable dihedral angles.*

If the ratio of the number of target distances to the number of rotatable bonds exceeds this value then the current loop generation attempt is abandoned and the process for generating a new loop is restarted.

Default: 1.1

arg8 *Maximum dihedral angle change.*

Tweaking a conformation involves iteratively adjusting the dihedral angles. The size of the changes in the dihedral angles for any one adjustment are scaled so that the largest change is less than arg7.

Default: 5 degrees

1 Initial construction parameters

arg2 *The number of attempts at generating a given dihedral angle before retreating to the previous rotatable bond.*

Default: 10

arg3 *The maximum total number of failed attempts at generating a new loop conformation.*

If this number is exceeded then the loop construction process fails and MacroModel terminates with an error message.

Default: 100,000

2 Geometric parameters

arg5 *The closest permissible approach of a loop heavy atom to a heavy atom within the rest of the protein.*

Default: 2 Å

arg6 *The closest permissible approach of a loop heavy atom with another loop heavy atom.*

Default: 2 Å

Related DEBG flags: 555

ARPK — ARPacK parameters

This opcode applies to LMC2, VBR2, and LTNCG (MINI arg1=11)

The ARPACK package was developed by Danny Sorensen, Richard Lehoucq, Chao Yang, and Kristi Maschhoff at Rice University [33, 34]. Instead of attempting to solve a huge eigenproblem directly, ARPACK solves it indirectly by solving a series of small problems of much lower dimensionality and references the huge matrix only implicitly through its product with some vectors. ARPACK is ideally suited for LLMOD calculations, because direct calculation of the low-mode eigenvectors of a huge Hessian matrix of a protein molecule is prohibitive. Instead, LLMOD utilizes ARPACK to compute the low-mode eigenvectors indirectly. The ARPK command allows a user to fine tune ARPACK parameters. Note that the VBR2 command also uses ARPACK to calculate vibrational modes.

DEBG 999 can be used to provide greater verbosity for ARPK.

arg1 *Select a method to calculate Hessian × vector*

0 Default: The Hessian is calculated analytically, but it is stored in a sparse vector representation, not in matrix form, and allows for fast matrix-vector multiplication, which scales only linearly with the number of dimensions. *This is the recommended method.*

1 $H \times v$ is calculated by a finite difference formula based on gradients (see Kolossváry and Keseru [29]). This method does not need second derivatives. It is fast, but is often unstable and converges only very slowly.

2 *H×v* is calculated using a LBFGS (see `MINI` arg1=10) representation of an approximation to the inverse Hessian matrix. This method is the fastest, but it can only be used to calculate the three lowest modes. Higher modes are usually only computational artifacts: they are totally useless for LLMOD. *Check arg4 if you choose this method*.

3 This method is a variant of arg1=1 using spectral transformation to boost convergence. Extremely slow, *not* recommended. If you choose to try it, check arg6 and arg7.

4 Same as arg1=0, but sparse Hessian storage is built on top of an explicit Hessian matrix. It is faster than arg1=0, but explicit Hessian storage makes it prohibitive for more than approx. 1000 atoms. (The Hessian storage scheme is the same used for TNCG minimization (`MINI` arg1=9).

5 *H×v* is calculated with *exact* arithmetic. The Hessian is not stored in any form, its elements are always recalculated as they are needed for computing $v_k(i)=H(i,j)*v_l(j)$ and $v_k(j)=H(i,j)*v_l(i)$. This is the most accurate method, but it is prohibitively slow on large molecules because of the burden of recalculating the Hessian numerous times.

arg2 *Dimension of "small" problems to solve (length of Arnoldi factorization)*

0 Default: 300

>0 Other value to be used. arg2 >= requested number of modes + 1 (`LMC2` arg3, `VBR2` arg2), recommended value: arg2 = 10 * requested number of modes

arg3 *Number of "small" problems to solve (number of Arnoldi iterations)*

0 Default: 450

>0 Other value to be used (maximum 25,000).

arg4 *This argument is multipurpose.*

If arg1=2, then arg4 specifies the number of *extra* LBFGS steps applied after minimization has reached convergence, to build a more accurate LBFGS representation of the Hessian matrix.

0 Default: 500

>0 Other value to be used.

If arg1=0, then arg4 is ignored. The amount of memory used for sparse Hessian is increased automatically as needed.

If arg1 is not 0 or 2, then arg4 is used to specify the amount of memory used for sparse Hessian storage.

0 Default: 20 (MB)

>0 Other value to be used.

Note that the user-defined memory is evenly distributed between storage of Hessian values and storage of corresponding (*i*, *j*) indices.

arg5 *Accuracy of ARPACK calculation in terms of ABS($\lambda_c - \lambda_t$) < arg5* ABS(λ_c) whereλ_c is a calculated eigenvalue and λ_t is the true eigenvalue.*

0 Default: 0.0001

<0 Machine precision (double precision).

>0 Other value to be used.

arg6 *Eigenvalue shift of spectral transformation. Used with arg1=3.*

0 Default: 5.0

>0 Other value to be used.

Spectral transformation transforms the eigenvalue spectrum such that the closest eigenvalue to arg5 becomes the largest eigenvalue and therefore, most likely the first eigenvalue to be found by ARPACK.

arg7 *Conjugate gradient tolerance criterion for spectral transformation. Used with arg1=3.*

0 Default: 0.01

>0 Other value to be used.

Spectral transformation involves minimum residual conjugate gradient iteration for which arg7 sets the convergence limit.

arg8 *Sparse Hessian cutoff value. Used with arg1=0 and arg1=4. (Similar to* MINI*/arg6).*

0 Default: 0.001

>0 Other value to be used.

Used with arg1=0, any H_{ij} term that is less than arg8 in magnitude is excluded from the sparse Hessian representation. A single H_{ij} element is the sum of a number of H_{ij} terms coming from different interactions that map to the same coordinate indices (i, j). On the other hand, used with arg1=4, arg8 is interpreted the same way as arg6 in the MINI command, i.e., only those elements of the Hessian matrix that are greater than arg8 are used in the sparse Hessian representation.

MCMM — Monte Carlo Multiple Minimum

MCMM [35, 36] is our recommended conformational search method. The input structure will be modified by random changes in torsion angles and/or molecular position as specified by the TORS or MOLS commands. Ordinarily, whether a single structure or multiple structures appear in the input file, they will first all be read in, minimized and treated as if already found by the MCMM procedure. This allows a new search to be initialized from the output of a previous search, by using the output file of the old search as input for the new one. However, if the necessary READ and MINI commands are placed within a BGIN/END loop, then a separate search is carried out for each input structure. This is called a *serial search*. For such serial calculations, the AUTO opcode is also needed.

The TORS command is used to specify dihedral angles to be varied; the MOLS command specifies relative positions of multiple molecules, as in an enzyme-substrate docking procedure. In addition, RCA4 commands and LIGB commands can be used to open rings and break ligand bonds, respectively, before performing torsional or relative molecular motion. CHIG commands should be specified for ring-closure atoms as well as to retain chirality about other centers. TORC commands may be used to hold double-bond configurations constant.

The AUTO opcode can be used for automatic setup of any kind of MCMM calculation.

Only unique structures will be retained, as in the MULT conformational searches. Use DEMX to set an energetic window to select low energy conformations. It is usually found that not all structures converge in minimization during a conformational search. A MULT minimization of the output file is recommended to achieve convergence for the final result.

While the default search method is random walk, we find that the usage-directed search (MCSS arg1=2) gives improved search performance.

arg1 *Number of Monte Carlo steps to be carried out before stopping*

0 Carry out the search until arg2 structures have been found. We do not recommend the use of this default.

arg2 *Maximum number of structures to retain while running*

0 The maximum number of conformations to retain while running is the value specified in arg1 or 10,000, whichever is greater.

>0 This is the maximum number of conformations to retain while running.

arg3 *Number of variables altered in each step*

The number of torsional angles varied and/or molecules moved in each step (default: 3). The value specified is randomly varied by +/–1 to prevent concentration of the search in local areas of conformational space. The default (3) thus alters 2–4 variables at each step.

Greater control over this parameter may be obtained using the `MCNV` command; this is the most common procedure.

arg4 *Search mode*

0 Global search (each Monte Carlo step begins with the preceding Monte Carlo structure, providing the structure is within 100 kJ of the global minimum). This is equivalent to `MCSS` (arg1=0, arg2=0, arg4=100.0).

1 Local search (each Monte Carlo step begins with the original structure). Generally used with small coordinate variations in `TORS` or `MOLS` to find other minima which are closely related to the starting structure.

arg6 *Allowable interatomic approach distance*

Fraction of sum of van der Waals radii which is used as a closest atomic approach limit (default: 0.25).

`SPMC` — Systematic Pseudo-Monte Carlo search

Similar to `MCMM`, but invokes systematic search in place of random search. The search begins at low torsional resolution (120˚), searches all angles without duplicating coverage, then doubles the resolution, etc. This method has the advantage of not retracing its path and consequently converges the final stages of the conformational search more efficiently than `MCMM`. Like `MCMM`, the method is effectively open-ended: it will search conformational space until stopped by the user or with arg1.

SPMC searches are conducted starting from a single input structure. However, if the necessary READ and MINI opcodes are placed within a BGIN/END loop, then a separate search is carried out ion each input structure. This is called a *serial search*. AUTO may be used to set up single SPMC searches and is required for serial SPMC searches.

It is suggested that torsional memory (MCSS arg3) be activated when using SPMC to prevent retracing of points in conformational space when starting from different starting geometries. If rings are being varied (i.e., RCA4 commands are being used), geometrical preoptimization (MCOP arg2) should also be activated.

Use with MCNV arg1=1 and arg2=N–1, where N is the number of variable torsions plus the number of molecules being independently translated/rotated with MOLS commands.

Details of the method are provided by Goodman and Still [37].

The arguments are the same as for MCMM, except arg3:

arg3 *Maximum resolution for torsional alterations*

Default: 24, implying angular resolution of 360˚ / 24 = 15˚.

MCOP — Monte Carlo OPtions

This command alters the data written to the .log file, and also the geometrical optimization routine. If MCOP is omitted, this is equivalent to setting arg1=250 and arg2=0.

Note: Despite its name, MCOP specifies parameters for use in low-mode (LMCS and LMC2) and LOOP calculations, as well as Monte Carlo searches. Arguments 4 and 5 only apply to LMCS and LMC2 jobs. Starting in MacroModel 6.5, these arguments have new meanings and arguments 6 and 7, which pertained to the earlier LMCS methodology, have been eliminated.

arg1 *Number of steps between printout to log-file*

0 Print to log file every 250 Monte Carlo steps.

1 Print to log file every step.

n Print to log file every *n* steps.

arg2 *Geometrical preoptimization*

0 Off.

1 On. Preoptimizes variable internal coordinates to improve ring closure distances. Recommended for SPMC of ring systems.

arg3 *Frequency of updating a conformational search*

When an update is performed, the following actions take place:

- The current `.tmp` file is removed, and structures that are to be saved (based on the value of the current global energetic minimum) are written to the output file.
- A summary of the progress of the search so far is written to the log file.
- If a *jobname*`.upt` file is created in the directory from which the job was initiated, an update is performed immediately, in addition to those specified by arg3.

0 Default: Perform an update every tenth of a run, but not more often than every ten steps nor less often than every 500 steps.

n Perform an update every *n* steps.

arg4 *LMCS serial job*

0 Not an `LMCS` serial job. A single low-mode search is performed, with all input structures treated as seed structures, as if found in previous iterations.

Note: arg4=0 is unsuitable for non-conformers.

≠0 `LMCS` serial job; this implies that a separate conformational search will be performed for each structure in the input file. This takes advantage of the ability of `LMCS` to define fruitful search directions without specification of variable torsions. An `LMCS` serial job can be run only when there are no commands specifying atom numbers—such as `TORS` or `CHIG`—which might translate to incorrect specifications in the different input structures.

arg5 *Probability of taking a TORS/MOLS step*

0 If this is an `LMCS` job, all steps will be LMCS steps.

n If this is an `LMCS` job and there are `TORS` or `MOLS` commands present, this fraction of moves will be `TORS` or `MOLS` (i.e., not LMCS) moves.

arg6 *Maximum number of conformers to write to the output structure file for each search performed*

For example, in a serial search, the search for each input structure would produce at most the number of structures specified here.

Using a value other than 0 or 1 for this argument is not permitted in a `ligprep` licensed calculation where only one output structure is permitted per structure processed.

0 Use setting from the previous MCOP command.

If no previous MCOP command has been issued, use the number of structures to save internally, which is controlled by arg2 of the search command (MCMM, LMCS, LMC2).

>0 Save up to this many structures per search.

arg7 *The number of conformers read in from the input structure file to seed each search in a serial search.*

This argument is primarily intended for use in MBAE conformational searches when ALGN has been used to produce multiple orientations for ligand structures.

Do not use this argument in LOOP calculations.

Do not use this argument in ligprep licensed calculations.

0, 1 Seed each search with one structure.

>1 Seed each search with this many input structures.

MCSS — Monte Carlo Structure Selection

This command allows the program to select starting geometries for Monte Carlo search steps in several different ways. Arg1 selects between random walk and two usage-related criteria. Arg2 affects weighting among selected structures for low energy geometries. Arg5 gives an optional energy window which prevents structures which are high in energy from being chosen.

This command is active only when doing global searching (i.e., when MCMM or SPMC arg4 = 0). In any case, structures must be within 100 kJ of the current global minimum to be candidates for starting geometry selection.

arg1 *Starting structure selection criterion*

0 Random walk. Most recent structure will be chosen whose energies are allowed by args2 and 5.

1 Use-directed. The least used structures will be used as starting geometries if their energies are allowed by args2 and 5. “Use” is defined as (times used as starting structure) – (times resulting structure is kept).

2 Use-directed. The least used structures will be used as starting geometries if their energies are allowed by args2 and 5. Use defined simply as (times used as starting structure).

arg2 *Energetic window modifier*

0 Arg5 will be used directly; recommended.

1 Arg5 will be multiplied by a random number between 0 and 1.

arg3 *Torsional memory selection*

0 Torsional memory is not used.

>0 Torsional memory is used. Structures are considered identical if all torsions match within RES/arg3 where RES is the operative search resolution (smallest value = 2).

arg5 *Energetic window, kJ/mol*

Current structure will be used as a starting geometry for a subsequent step only if its energy is within arg5 kJ/mol of the lowest energy structure yet found. A good choice for arg5 is simply the value of the overall energetic window being used in DEMX. Default: 100 kJ/mol.

FLAP — use FLAP algorithm for ring conformations

Use the FLAP ring vertex reflection algorithm [53] to generate ring conformations in MCMM, low-mode, or mixed MCMM/low-mode conformational searches. This opcode must be added before the relevant MCMM or LMCS opcode.

arg5 *Probability of using a FLAP step*

Specify the fraction of conformational search steps that will use FLAP. If used in conjunction with MCRC, the sum of the FLAP and template probabilities must not be greater than 1.

0 Use the default value of 0.5

>0 Use this value. Should be between 0 and 1.

MCRC — Monte Carlo Ring Conformations

Use templates of ring conformations in MCMM, low-mode, or mixed MCMM/low-mode conformational searches. This opcode must be added before the relevant MCMM or LMCS opcode.

arg1 *Use of templates*

0 Use templates in conformational search (default)

−1 Do not use templates in conformational search

arg5 *Probability of using templates*

Specify the fraction of conformational search steps that will use templates. If used in conjunction with FLAP, the sum of the FLAP and template probabilities must not be greater than 1.

0 Use the default value of 0.5

>0 Use this value. Should be between 0 and 1.

MCTS — Monte Carlo Torsion Selection

Experience has shown that this command is not terribly useful.

Use in conjunction with MCMM to favor torsion angle selection near local torsional minima for the angles being varied. One MCTS command is required for each torsion which is to be effected. Arg1-4 are the atoms defining a particular torsion angle (i.e., arg2-3 should appear as a rotatable bond in a TORS command). Arg5-7 are 1-fold, 2-fold, and 3-fold torsional barriers which are used to compute local torsional energies as part of the test for an allowable value of the randomly selected Monte Carlo angular change. After a Monte Carlo variation of a torsion angle described by arg1-4 of this command is performed, a local torsional energy (E_T) is computed based on the value of the dihedral angle, τ, using the usual MM2 formula:

$$E_T = (V_1/2)(1+\cos\tau) + (V_2/2)(1-\cos 2\tau) + (V_3/2)(1+\cos 3\tau) - E_{T,min}$$

where $E_{T,min}$ is the minimum possible value of E_T. If E_T is greater than 1, the torsion is rejected and a new random torsion angle is chosen. Otherwise, E_T is compared with a random number between 0 and 1 and, if E_T is larger than that number, a new random torsion angle is chosen. This scheme selects for local torsions which are low in energy. For sp3-sp3 linkages, one can favor the gauche and anti conformers (the minima) by using a V_1 (arg5) and V_3 (arg7)of 0.25. This will cause totally eclipsed (0-degree torsion) torsions to be strongly disfavored, 120-degree torsions to be moderately disfavored and the gauche and anti conformations to be favored. By choosing V_1-V_3 with care and using a random number for comparison, even high energy geometries are occasionally explored.

arg1-4 *Atoms defining the torsion*

arg5 *V_1 (positive number gives anti minima)*

arg6 *V_2 (positive number gives eclipsed minima)*

arg7 *V_3 (positive number gives staggered minima)*

MCSM — Monte Carlo Single Minimum

This procedure is similar to that described by Li and Scheraga [38] and is a global search for the single lowest energy structure. The search can be conducted with or without minimization (depending on arg3). While this method is good at finding a single low energy conformer, there is no guarantee that it will locate the true global minimum energy conformer.

Monte Carlo internal coordinate conformational search is performed with minimizations on a single structure. Variable internal coordinates are specified by TORS and/or MOLS.

MCSM can be used with cyclic structures providing that either ring bonds are not varied or that ring closure commands (RCA4) are used for each ring in which variable torsions are used. During the run, random variations will be applied to 2-4 randomly selected dihedral angles from the TORS lists or molecular translations/rotations from the MOLS lists for each Monte Carlo step. If some other range of variable coordinates is desired, the range is set with the MCNV command.

If used within a BGIN/END loop, all structures in the input file will be read and the global minimum found will be listed to the output file at the end of every MC iteration set for each input structure. The ultimate global minimum would be found by examining the energies of each of the output structures.

Unlike MCMM, MCSM requires no MINI command. It must appear after a READ command.

In general, Monte Carlo searches should use CHIG commands to maintain all chiral centers and TORC commands to hold double bond geometries constant. If chirality or double-bond geometry is lost in any step, then following steps will be wrong if the resulting geometry is used for subsequent steps.

arg1 *Minimization mode*

0 Steepest descent.

1 PR conjugate gradient (best general method).

3 Variable Metric (not recommended with MCSM).

4 Full matrix NR (not recommended with MCSM).

9 TNCG (good for flexible structures).

arg2 *Line-search control for second-derivative methods (arg1=4 or 9)*

0 No line searching (best choice).

1 Line searching on.

arg3 *Maximum number of minimization iterations*

arg4 *Number of Monte Carlo cycles (default: 100)*

arg5 *Initial temperature (K)*

0.0 Metropolis sampling will not be done; every structure will be used as the next starting point

arg6 *Final Temperature (K)*

0.0 Continuous sampling at the arg5 temperature will be done throughout the run. If a nonzero temperature is supplied, cooling from the arg5 to the arg6 temperature will be carried out during the run. This slow cooling is equivalent to simulated annealing.

arg7 *Step size buffer*

As in `MINI` arg5.

arg8 *TNCG Hessian cutoff*

As in `MINI` arg6.

`MCNV` — Monte Carlo Number of Variables

This command resets the degrees of freedom (number of torsion angles varied plus number of molecules moved in space) altered in a single search step. If this command is not used, the value is taken from arg3 of the `MCMM` command. This command overrides the `MCMM` arg3.

For `MCMM` on single unsymmetrical molecules, we find it best to specify the range as 1 to *N*, where *N* is the number of variable dihedral angles, when `TORS` commands are being used. It is best to use a range of values, rather than a single value. When `MOLS` is being used, *N* should be incremented by one. When `ZMAT` is being used, *N* should take account of all degrees of freedom.

Used with `MCMM`, `MCSD`, `MCLO`, `IMPS`, and `SPMC` commands.

In the context of MC or MC(JBW) simulations this command sets the number of degrees of freedom to be changed at each MC step or randomized at each MC(JBW) step. A degree of freedom is either a bond length, bond angle or torsion or a molecular translation or rotation along or about a single axis. This number should be set in such a way to provide a compromise between acceptance rate and conformational interconversions. When args 1 and 2 of this command differ, MCNV defines a range for the number of degrees of freedom to be changed at each step. When the two arguments are identical, MCNV defines an exact number for the degrees of freedom to be changed at each step. In both cases, the initial values will be modified during the run by the adaptive mechanism unless debug 103 is defined.

For MC, MC(JBW) or MCSD simulations, a number of degrees of freedom must be defined and consequently, the MCNV command should be present; however, its two arguments can be 0, in which case the program will provide a default range, from 1 to maximum number of the degrees of freedom of the molecules. Such a range is probably not efficient. For MC(JBW)/SD, no randomization in the JBW part is needed and all the randomization can effectively be done by the SD part of the simulation. Doing so will increase the number of conformational interconversions. In order to achieve that, the MCNV command should be omitted and arg 2 of the MCSD command should be set to a negative number.

arg1 *Minimum number of degrees of freedom altered*

0 Default: 1

arg2 *Maximum number of degrees of freedom changed in a MC step*

0 Default: number of variable degrees of freedom in the system.

arg4 *Cluster torsions varied at one time*

This is no longer seen as a useful option.

0 Do not cluster torsions; recommended.

1 All torsions rotated will be in a contiguous group as defined by the ordering of torsions in the TORS commands.

2 Allow a single intervening unused torsion.

n If, at a given stage, *m* torsions are rotated, these are selected from a contiguous group of (m+n–1) torsions selected from the list taken from the TORS commands.

SEED — random number generator SEED

This command sets a starting value for the random number generator. Used to start the Monte Carlo random number generator at a different point so that repeated Monte Carlo or molecular dynamics runs give different results. The starting value should not exceed 700 000; if it does it will be divided by 2 until it is less than 700 000.

arg1	*Seed value*
–1	Use a value generated from the time and date.
0	Use the default value.
>0	Use the value specified (must be less than 700 000), or use positive values less than 78593 if the MacroModel random-number generator is used (DEBG 178).

TORS — variable TORSion selection

Each TORS command specifies up to two torsions, using the numbers of the two central atoms. These will be used as variable dihedral angles by the MCMM, SPMC or MCSM commands. The actual number of torsions which will be varied during a single Monte Carlo step depends on the search method, but the number varied will be taken from the list specified in TORS commands. A given random torsional variation will be plus or minus a random number selected from the range extending from the arg5 to the arg6 specification.

Variable torsions within rings require the ring-closure commands RCA4 in addition to TORS commands. It is advisable to specify at least two variable torsions within each ring containing RCA4 ring closures. The atoms of a ring closures (args 2 and 3 in RCA4) must not be listed as variable torsions.

The minimum and maximum angular increment (arg5 and arg6) refer to the torsions given in arg1-4. It is possible to use a different angular increment for each torsion by using only arg1, arg2, arg5, and arg6 and a different TORS command for each torsion. For global searching, arg5 and arg6 of 0.0° and 180.0° are appropriate values. If you wish to focus the search on conformations having only small angular variations from the starting conformation, a value of 30.0° for arg6 could be used.

If you are searching multicyclic ring systems, you should include CHIG commands for any substituted atoms at the ring closure atoms (arg2 and arg3 in RCA4) to assure maintenance of stereochemistry.

When doing substructure MC searching, always order the pairs of atoms defining torsions such that the second atom of each pair is not connected to any fixed atoms (FXAT) except via the first atom (in the torsional movements, the chain connected to the second atom is the one that

will actually be moved). If both ends are anchored by FXAT commands, then a ring closure (RCA4) command will be necessary. TORS and MOLS commands can be used together.

arg1-2 *Atom numbers specifying first torsion*

arg3-4 *Atom numbers specifying second torsion*

arg5 *Minimum dihedral angle variation*

A positive number in degrees (default: 0.0˚).

arg6 *Maximum dihedral angle variation*

A positive number in degrees (default: 180.0˚).

TRES — Torsional RESolution

This command may be used in the SPMC systematic pseudo Monte Carlo searches to alter the initial resolution of the search around a particular torsion angle. This command must come after the TORS command.

arg1-2 *Atoms defining the torsional angle*

These must have been listed already with a TORS command.

arg3 *Resolution for this angle*

The value in degrees of the initial moves will be 360˚/arg3; thus arg3=3 (the default) gives 120˚ resolution.

MOLS — variable MOLeculeS selection

This command selects molecules to be independently rotated and/or translated during a conformational search. It is used for configurational/conformational searches of complexes. In particular, given a docked bimolecular complex, MOLS can be used to translate and rotate the smaller molecule within the binding site of the larger one in order to explore possible binding geometries. MOLS and TORS commands can be used together; this allows the internal geometry of the separate molecules to be explored together with the relative orientation. If there are N molecules in the system, it suffices to specify N–1 of them in MOLS commands to ensure that all are adjusted.

MOLS commands must come after TORS commands.

During a search, random molecules are selected for motion. For each molecule, rotation about all three axes and translation along all three axes are performed by amounts selected randomly

from within the ranges specified in arg5-6 and arg7-8. MOLS has the same relationship to LIGB as TORS has to RCA4: the bonds specified in a LIGB command are broken before the MOLS-specified molecular motion is carried out, then the LIGB bonds are remade prior to minimization of the resulting structure.

Note: During a long enough search, if pair-list cutoffs are in effect, as is normal (see EXNB), one molecule may eventually wander far enough away from another that no nonbonded energies between them exist anymore. In this situation, further searching just explores random spatial dispositions of this pair with no energetic contribution from their mutual interaction. Unless already-found binding conformations are lower in energy by at least the DEMX-specified energy, this will lead to essentially an infinite and fruitless search of conformational space. To avoid this, use FXDI to constrain the distance between atom pairs spanning pairs of molecules to some maximum distance. This maximum distance should be specified as the half-width of an FXDI potential. In practice, we do this only if we encounter difficulties without doing so.

arg1-4 *Atom in a molecule to be moved*

Each nonzero atom given specifies independent motion of the entire molecule containing the atom. Thus, to specify independent motion of two molecules, put an atom number from the first in arg1 and an atom number from the second in arg2.

<0 Perform rotations about the atom number given.

>0 Perform rotations about the center of mass of the molecule containing the atom.

arg5 *Minimum rotational variation*

Angle in degrees (default: 0.0).

arg6 *Maximum rotational variation*

Angle in degrees (default: 0.0, i.e., no rotation). 180° is a reasonable value.

arg7 *Minimum translational variation*

Movement in Angstroms (default: 0.0).

arg8 *Maximum translational variation*

Movement in Angstroms (default: 0.0). A value in the range 3-5 Å is reasonable value.

LIGB — LIGand Bonds

Used chiefly for configurational searches of inorganic complexes in conjunction with the VDWB command.

This command defines bonds to be broken, creating molecular fragments which will be moved independently during the search. A MOLS command must be present for each fragment to be moved. For a bidentate ligand, there will be two LIGB bonds specified for the single MOLS command that moves the ligand; for a tridentate ligand, there will be three, and so on.

This procedure allows an MCMM search to find, for example, both *mer* and *fac* isomers of an octahedral complex with stoichiometry MA_3B_3. LIGB can also be used to extend a conformational search to a configurational search; for example, by specifying bonds to chiral carbons as LIGB bonds, the R and S configurations about these carbons will be explored.

arg1 *First atom in bond to be broken (typically a metal atom)*

arg2 *Bond(s) to be broken*

0 Add all bonds to arg1 to the LIGB list.

>0 Add the arg1-arg2 bond to the LIGB list for arg1.

<0 Remove the bond between arg1 and |arg2| from the LIGB list for arg1, if such a list already exists, or, if no such lists exists, create one with all bonds to arg1 on it except this one.

RCA4 — Ring Closure Atoms

This command directs the program to temporarily break a bond to sever a ring for the purpose of Monte Carlo torsion angle searching. The first 4 arguments are four atoms within the ring which comprise a ring-closure torsion angle. One RCA4 command is necessary for each ring with dihedral angles specified in a TORS command. Used with MCMM, SPMC, IMPS, and MCSM.

With MCMM and MCSM, and for small-medium rings, arg5 and arg6 should be approximately 0.5 and 2.5. For large rings, arg5 and arg6 should be ca 0.1 and 5.0.

With SPMC and for small-medium rings, arg5 and arg6 should be approximately 1.0 and 2.0. For larger rings, arg5 and arg6 should be ca 0.5 and 3.0-4.0. The exact choice is not very important but has a minor effect on search efficiency.

It is forbidden that an arg2 or arg3 atom be used in more than one RCA4 command (common ring closure atoms are not allowed). The closure angle arguments (arg7 and arg8) are optional and refer to both closure angles 1-2-3 and 2-3-4. Our tests so far indicate that there is little reason to use arg7 and arg8 closure angle constraints.

arg1-4 *Atom numbers within ring*

Four contiguous atoms within a ring. The closure bond is the one between arg2 and arg3.

arg5 *Minimum allowable ring closure distance (Å) (default: 0)*

arg6 *Maximum allowable ring closure distance (Å)*

The default is essentially infinite, so that ring-closure distance never precludes minimization on a structure produced by Monte Carlo variations.

arg7 *Minimum allowable closure angle (˚)*

Default = 0.0

arg8 *Maximum allowable closure angle (˚)*

The default is 180˚, so that ring-closure angle never precludes minimization on a structure produced by Monte Carlo variations.

DISC — DIStance Check

This command causes the distance between atoms given in arg1 and arg2 to be monitored and structures which have such distances outside the allowable range to be eliminated from the output file. Used only with MULT, MCSM, and MCMM.

arg1-2 *Atom numbers defining distance*

arg4 *Control*

0 Check distance before minimization.

1 Check distance after minimization.

arg5 *Minimum allowable distance (Å)*

arg6 *Maximum allowable distance (Å)*

TORC — TORsional Check

This command causes the absolute value of the torsion angle defined by arg1-4 to be monitored and structures which have such torsions outside the allowable range to be eliminated from the output file. Used only with MULT, MCSM, and MCMM. The test is applied after minimization in each case, and acts as a filter, particularly during conformational searches.

This command is intended to reject unwanted cis or trans isomers created during a conformation search. Appropriate values for rejecting all trans isomers of a cis torsion (original torsion angle ~0) are arg5=0 and arg6=90. To reject all cis isomers of a trans torsion (original torsion angle ~180), use arg5=90, and arg6=180.

Note: Torsion checks do not limit the range of conformers searched or constrain the torsion during minimization. See FXTA on page 71 for details on applying a torsional constraint during minimization.

arg1-4 *Atom numbers specifying a dihedral angle*

arg5 *Minimum allowable angle (\degrees\)*

arg6 *Maximum allowable angle (\degrees\)*

MCMF — Monte Carlo Maximum constraint Failures

This command allows you to adjust the maximum number of constraint failures, e.g. ring closures (RCA4) or constrained torsion (TORC), that are allowed before accepting a faulty structure. A limit of some kind is appropriate since the user could supply constraints that make valid structure generation impossible. Thus the program tries a given starting geometry and a given number of varying torsions (or other coordinates) repeatedly until arg1 failures (or the default of 10000) have occurred. Then, the program allows a different number of varying torsions and makes arg1 new attempts to create a valid structure. This process is repeated arg2 times before a faulty structure is accepted. Thus when constraint tests have failed arg1 times, the program allows arg2 failures (of arg1 tries each) before accepting the structure anyway. Such faulty structures often fail to minimize properly.

arg1 *Maximum number of constraint failures with constant torsion set (default: 10000)*

arg2 *Maximum number of attempts with variable torsion sets (default: 10)*

SMPL — monte carlo SaMPLing

This command allows one to write sample structures to disk during a Monte Carlo Single Minimum (MCSM) run.

arg1 *Sampling interval*

0 Write the last structure sampled and the global minimum; the latter appears last in the output file.

n Write every *n*th structure sampled.

SRNO — SeRial Number Ordering

arg1 *Number to be added to serial number property in output structure file*

4.11 Molecular Dynamics and Monte Carlo Simulations

MDYN — Molecular DYNamics

Carry out molecular dynamics. Structures must be fully minimized before molecular dynamics is initiated and initial conditions (temperature, time step, and total time) must be selected. Temperature maintenance with a constant temperature bath is recommended. If too high a temperature or too large a time step is selected, the structure will eventually gain too much kinetic energy and blow up (program automatically aborts if the kinetic energy/atom exceeds 100 kJ/mol). The structure may also blow up if it is not sufficiently minimized and energy is added (e.g., MDIT). If this occurs, reduce the time step, turn on SHAKE (arg 2) and try again.

During the run, average temperatures (both over the most recent 0.5 ps and over the entire simulation so far), the instantaneous total energy (potential plus kinetic energy) and the average enthalpy over the entire simulation (taken as the average steric energy) are written to the log file. If you choose the full printing option (arg1=2), the information is written to the .mmo file as well. If you are using constant energy molecular dynamics (no temperature control), the energy may change over the course of the simulation: if the temperature listed after 0.5 ps or so is more than 5% too high, then it is likely that the starting geometry was not completely minimized or that a lower energy nearby minimum has been found.

If you are not using temperature control and wish to readjust the temperature, stop the batch job and set the desired temperature with arg 7. Start the run again and the temperature will adjust itself over the first several ps.

It is best to do the simulation in stages (e.g., to begin by equilibrating the system prior the actual run) having a MDYN command for each stage. If the MDRE (reset) command is not used, the run will continue from the previous end point. Equilibration runs are suggested for good thermodynamics especially if there is no reason to believe that the starting structure is the fully minimized global minimum. We suggest at least 5-50 ps of equilibration before data collection for all molecular dynamics operations other than conformation searching where equilibration is unnecessary. Experience shows that ca .2-.5 times the number of heavy atoms gives a reasonable equilibration time in picoseconds.

The time step (in femtoseconds (Default = 1.5 fs), arg5) is the time interval of each step of the molecular dynamics run. Energy stability is best at or below 1.0 fs but time steps up to 2.5 fs may be used to reduce the overall time required for the simulation. If too large a time step is

used, the structure will gain energy each step and eventually blow up (simulation will abort). In general, higher temperatures (kinetic energy) require smaller timesteps. If mediocre energy stability is acceptable (e.g., for local conformation searching), then an initial temperature of 300-400 K with a timestep of 1.7-2.0 fs provides a good compromise of speed and stability if SHAKE (constrains H bond lengths) is used. If precise energies are important, a timestep of 1.0 fs with temperature control (see arg 7), or a timestep of 1.5 fs with SHAKE (arg 2) and temperature control (arg 7) is recommended. Molecules with explicit hydrogens (e.g., when using MM2) may require a smaller timestep and/or use of SHAKE (see arg 2) to constrain bonds to hydrogen.

At the end of each `MDYN` run, the final structure must be written to the output file using the `WRIT` command, otherwise no structure will be saved in the output file (unless `MDSA` is used). It is often useful to save final velocities using the `MDVE` command so that subsequent runs can start with equilibrated structure (read the final structure and the corresponding velocity file for the next run). Multiple `MDYN` commands within the same MacroModel run need not be accompanied with `WRIT` or `MDVE` commands.

`DEBG 1` gives energy output to the log file ten times as often as it otherwise would.

arg1 *Extent of listing*

0 No listing to `.mmo` file. Summary of results in `.log` file (Default).

1 Initial and final potential and kinetic energy, results of distance and angle monitoring, average enthalpy <H>.

2 Potential and kinetic energy results of distance and angle monitoring for each 10 timesteps.

arg2 *SHAKE protocol*

SHAKE is taken from Ryckaert [39]. We do not recommend using SHAKE in `MCSD` runs, as it can result in less than optimal temperature control.

0 SHAKE inoperative (default).

1 Bonds to H constrained.

2 All bond lengths constrained.

SHAKE cannot be used with stretch-bend potentials, which will be turned off automatically whenever SHAKE is activated.

arg3 *Molecular or stochastic dynamics selection*

0 Molecular dynamics (default).

1 Stochastic dynamics. Temperature control is an integral part of stochastic dynamics and a nonzero temperature must supplied in arg7. With stochastic dynamics, a timestep of no more than 1.5 fs and SHAKE is recommended. Generates the canonical (constant-temperature) ensemble.

Total translational and rotational momentum is reset (zeroed out) by default when ordinary molecular dynamics is run, but not when stochastic dynamics is run. This behavior can be overridden using the `MDMT` command.

arg4 *Frequency of data collection*

Used to determine the time interval for computation of average enthalpy.

0 Default: 25 fs, but 5 fs for runs of less than 10 ps; increased for runs of more than 1000 ps.

n *n* fs

arg5 *Dynamics time step in femtoseconds (default 1.5 fs)*

arg6 *Total time of dynamics run*

0 Default: 1.0 ps.

>0 Interpreted as ps.

<0 Run 100 times the number of ps entered.

arg7 *Temperature (K)*

For constant temperature dynamics. Temperature is maintained by a thermal bath [40] with molecular dynamics. Temperature control is an integral part of stochastic dynamics [41]. A temperature is optional in molecular dynamics but necessary in stochastic dynamics. If no temperature is specified in molecular dynamics, then the simulation will be at constant energy rather than constant temperature. Constant temperature dynamics is recommended for most dynamics simulations. This argument is also used for the starting temperature in techniques in which the temperature is not held constant.

arg8 *Bath time constant*

For molecular dynamics, the default is 0.2 ps and represents a thermal bath time constant. For stochastic dynamics, the frictional coefficient gamma is equal to 1/(2tau).

MDSA — Molecular Dynamics structure SAmpling

Also used to specify structure sampling for Monte Carlo procedures.

Sampling of structures during molecular dynamics or Monte Carlo run by time. MDSA commands must come before the MDYN, MCSD or MCLO command from which the samples are to be taken.

arg1 *Total number of structures to be sampled*

This number of structures will be written to the output structure file over the course of the run. Default: sampling will be disabled, unless arg5 is specified.

arg2 *Frequency of importance-sampling population and statistics reporting*

Number of times during the run importance-sampling populations and overall statistics will be printed to the log file.

0 No reporting.

<0 Print this information is printed each time a dynamics summary line is printed to the log file.

arg3 *Frequency of importance-sampling transition reporting*

Number of times during the run importance-sampling transition statistics will be printed to the log file

0 No reporting.

<0 Print this information is printed each time a dynamics summary line is printed to the log file.

arg4 *Frequency of monitoring-statistics reporting*

Number of times during the run monitoring statistics (e.g., MHBD) will be printed to the log file. For all-degrees-of-freedom Monte Carlo (ZMAT), also controls output of additional statistics, such as acceptance ratio.

0 No reporting.

<0 Print this information is printed each time a dynamics summary line is printed to the log file.

arg5 *Time interval for writing of structure samples*

0 Ignore—use arg1, if nonzero.

>0 Time interval in ps.

arg6 *Time interval for writing summary lines to log file*

0 Program chooses a "reasonable" value.

>0 Time interval in ps.

arg7 *Control writing of output structure file at the start of each MDYN run*

Initially, MacroModel is set to not clear the output structure file at the start of each MDYN run. If arg7 of MDSA is set to a value greater than zero, the default behavior is turned off and the output structure file is wiped clean at the start of each subsequent MDYN run. Setting arg7 of MDSA to a value less than zero "switches off" the effect of having set arg7 to a value greater than zero in a previous MDSA line in the command file.

If arg7 is set to zero, then the currently set MDSA behavior will be used. For example, if arg7 had been set to a value greater than zero at an earlier point in the .com file, setting arg7 to zero in subsequent line would cause the previously set behavior (wipe the output structure file) to continue. On the other hand, if arg7 had been set to a value less than zero at an earlier point in the .com file, setting arg7 to zero in subsequent line would cause the previously set behavior (do not wipe the output structure file) to continue.

If the first MDSA line in the .com file is set to 0, then the previously set behavior (MacroModel's default behavior) of not wiping the output structure file would be used.

>0 Wipe the output structure file clean at the start of each MDYN run after this line in the .com file.

<0 Do not wipe the output structure file clean at the start of each MDYN run after this line in the .com file.

0 Do not change the previously set MDSA arg7 behavior.

MDMT — Molecular Dynamics MomenTum reset

Used to control translational and rotational (angular) momentum which will be periodically (0.1 ps by default) reset to zero. MDMT is operational by default (i.e., MDMT is unnecessary unless it has previously been turned off), and its normal mode is constraint of the total molecular system to have zero translational momentum.

Note: Structures viewed graphically will not appear to translate or rotate even if momentum is not zeroed because they are translated/rotated to the original position and overall orientation before writing to any output file.

The frequency of momentum zeroing is set by arg 5.

arg1 *Choice of momentum elements to be reset*

0 No momentum resetting. This is the default for stochastic dynamics.

1 Set total translational momentum to zero.

2 Set total translational and rotational momentum to zero. This is the default for molecular dynamics.

arg5 *Frequency of momentum resetting*

Default: 0.1 ps.

MDAR — Molecular Dynamics ARea monitoring

Atomic area monitoring during the molecular dynamics run. Output consists of histogram-like compilations of individual atomic areas and the averages. The output will be written to the mmo output file. If no probe radius is specified, then the areas will reflect the van der Waals surface areas for the atoms specified. If all atoms are to have the same probe radius, use MDAR commands loaded with four atoms at a time using args1-4.

Data is accumulated every 10 time steps throughout the simulation and also when the structure is sampled. DEBG switch 1 lists each value to the .log file. A maximum of 250 atoms may be monitored simultaneously.

This command must precede the MDYN command to be monitored.

arg1-4 *Atom numbers for area monitoring*

arg5 *Probe radius (Default: 0)*

MDAP — Molecular Dynamics Atom Position monitoring

Atom Position monitoring during the molecular dynamics run. Output consists of average rms deviation (Å) from original position. The output is written to the log and `.mmo` output file.

Data is accumulated every 10 time steps throughout the simulation and also when the structure is sampled. `DEBG` switch 1 lists each value to the `.log` file. A maximum of 250 atoms may be monitored simultaneously.

Note: This command must precede the `MDYN` command to be monitored.

arg1-4 *Atom numbers for position monitoring*

MDDI — Molecular Dynamics DIstance monitoring

Distance monitoring during a molecular dynamics run. Output consists of histogram-like compilations of internuclear distances/frequencies and the average distance. Histograms have 100 bins covering the range of arg6 to arg6+arg5*100. The default range is 0–10 Å in 0.1 Å increments. The output is written to the `.mmo` output file.

Data is accumulated every 10 time steps throughout the simulation and also when the structure is sampled. `DEBG 1` lists each value to the log file. A maximum of 100 distances may be monitored simultaneously.

This command must precede the `MDYN` command for monitoring to take place.

arg1 *First atom of pair*

0 No distances monitored.

>0 First atom of pair.

arg2 *Second atom of pair*

arg5 *Bin width for histograms*

Width of bins for distance monitoring in angstroms. The default value is 0.1 Å, and the minimum value is 0.01 Å.

0 Use the default value of 0.1 Å.

>0 Use this value.

arg6 *Lower limit of histogram range*

Smallest distance value recorded in the histogram, in angstroms. Default: 0 Å.

>0 Use this value.

MDBA — Molecular Dynamics Bond Angle monitoring

Bond Angle monitoring during the molecular dynamics run. Output consists of histogram-like compilations of angles/frequencies and the average angle. Default resolution for the histograms is 5 degrees (see arg5). The output will be written to the `.mmo` output file.

Data is accumulated every 10 time steps throughout the simulation and also when the structure is sampled. `DEBG 1` lists each value to the `.log` file. A maximum of 100 angles may be monitored simultaneously.

This command must precede the `MDYN` command for monitoring to take place.

arg1-3 *Atoms defining the bend*

arg5 *Resolution for reporting*

Resolution in degrees for the histogram listing to the `.mmo` file (default = 5 degrees). The minimum resolution is 1˚ and the maximum resolution is 180˚.

MDDA — Molecular Dynamics Dihedral Angle monitoring

Dihedral Angle monitoring during the molecular dynamics run. Output consists of histogram-like compilations of angles/frequencies and the average angle. Default resolution for the histograms is 10 degrees (see arg5). The output will be written to the `.mmo` output file (averages and histogram, printing must be on —see `MDYN` arg 1) and to the job log file (averages only). The average torsion angle cosine and the average cosine squared are also provided for use in Karplus-like coupling constant calculations.

Data is accumulated every 10 time steps throughout the simulation and also when the structure is sampled. `DEBG` switch 1 lists each value to the `.log` file. A maximum of 100 angles may be monitored simultaneously.

This command must precede the `MDYN` command to be monitored.

arg1-4 *Atoms defining dihedral angle*

arg5 *Resolution for reporting*

Resolution in degrees for the histogram listing to the `.mmo` file (default 10°). The minimum resolution is 1° and the maximum resolution is 360°.

MHBD — Monitor the occurrence of a Hydrogen BonD

This command allows monitoring of the number of times that distance and angle criteria are met during a molecular or stochastic dynamics simulation. The normal use of this command is to monitor the population of hydrogen bonds during a simulation. At each sampling point a distance and one or two angles are determined, and if these values meet criteria defined in the MHBD command line then a hydrogen bond is considered to exist. At the end of the simulation the occurrence of the specified hydrogen bond is reported to the log file as the percentage of structures in which the bond occurs.

arg1 *Donor atom (D)*

arg2 *Hydrogen atom (H)*

arg3 *Acceptor atom (A)*

arg4 *Atom bonded to acceptor (B)*

0 Default: not used.

Example:

```
                N —   H •••    O =   C
          arg:  1     2        3     4
abbreviation:   D     H        A     B
```

arg5 *Maximum allowable H–A distance*

0 2.5 Å

arg6 *Minimum allowable D–H–A angle*

0 120˚

arg7 *Minimum allowable H–A–B angle*

Used only if arg4 is nonzero.

0 90˚

MDAV — Molecular Dynamics coordinate AVeraging

Averaging of atomic Cartesian coordinates during the molecular dynamics run. The average atomic coordinates will be used as the structure coordinates at the end of the dynamics run. The coordinates will be written to the output structure file. To save these coordinates in the output file, it is necessary to use a WRIT command after the MDYN command.

This command must precede the MDYN run to be averaged.

arg1 *Control*

0 Perform no averaging.

1 Perform averaging.

MDIT — Molecular Dynamics Initial Temperature

Initial Temperature (K, Absolute) for the start of the run. This temperature is converted to the corresponding kinetic energy, and that energy is used to initialize the velocities of all atoms. A random Gaussian distribution of velocities is generated such that the total energy is distributed equally along the three Cartesian axes. 3RT is also supplied for translation/rotation for each molecule beyond the first. The total energy added is (3N-6M)*RT (M is the number of translation/rotation constraints, normally 1). Note that the MDIT command replaces any current velocities with random new ones and assumes that the structure is completely minimized (if it is not, the resulting temperature will be higher than that given in this command unless temperature control is used).

If no initial temperature is specified, the simulation keeps the kinetic energy from a previous run. If a structure is fully minimized and no initial temperature (or initial energy) is specified, the simulation will not run. If a structure is not fully minimized, the difference between the current strain energy and the minimum energy builds up in the kinetic energy term.

The standard dynamics protocol is to fully minimize a structure, then start the simulation either with the desired temperature, or slowly increase the kinetic energy (set an initial (MDIT) and warm the molecule over ca 5 ps (MDYN args 7 and 8)) prior to an equilibration run. If an equilibration period is used, the actual run will occur in a separate step without specification of new initial or final temperatures.

If a precise (harmonic) temperature is desired, use the MDYN arg7 to set constant temperature dynamics. This will insure that the final temperature attained will be that desired regardless of the relative energy of the starting structure and any nearby (local) minima after equilibration.

arg5 *Temperature (K)*

MDFT — Molecular Dynamics Final Temperature

Final Temperature (K, Absolute) to be achieved at end of run. The difference between the original bath temperature (MDYN arg7) and the final desired temperature (MDFT arg5) is slowly subtracted from the bath temperature over the course of the simulation. This temperature changing mechanism can be used to either heat or cool systems linearly during the simulation. The actual final temperature will be close but not equal to arg5. If a precise final temperature is desired, a subsequent MDYN run with the desired temperature in MDYN arg7 should be used. MDYN arg7 and arg8 can also be used to heat or cool molecules over the course of a simulation. This alternative would be effected by having an initial temperature as desired and the final temp as MDYN arg7 and weak coupling to the temperature bath via MDYN arg8.

arg5 *Temperature (K)*

0 MDFT inactive.

Values less than 0.0001 K are not allowed.

MDMC — Molecular Dynamics / Monte Carlo mixed mode

MCSD — Monte Carlo / Stochastic Dynamics mixed mode

These commands are synonyms; MCSD is the preferred form, since its name reminds the user that the mixed-mode procedure should always be carried out using stochastic dynamics. This is because stochastic dynamics and the Monte Carlo procedure both generate the canonical ensemble, whereas (constant-energy) molecular dynamics generates the microcanonical ensemble.

This procedure mixes stochastic dynamics and Monte Carlo (MC) internal coordinate movements. This makes it possible to explore conformational (phase, configurational) space more effectively than dynamics alone, while using the dynamics paradigm to obtain free energies. Internal coordinate Monte Carlo movements must be specified using MOLS and TORS commands as is done in MCMM conformational searching. TORS commands involving ring bonds and RCA4 commands are allowed with MCSD, but the acceptance ratio is very low. If this combination of commands is used, DEBG 104 should be given.

This command should only be used with stochastic dynamics (MDYN arg3=1 and arg7 !=0).

We do not recommend using SHAKE (controlled via arg2 of MDYN) during MC/SD simulations, as its use can result in less than optimal temperature control. Using MCSD without SHAKE sometimes requires a smaller simulation timestep than might otherwise be used. But since MCSD is very efficient at exploring conformational space relative to dynamics alone, this should not be an issue.

If the MC/SD simulation is unstable and fails, consider turning off long-range constant derivatives, by setting EXNB arg1 to 1, and increasing the non-bonded cutoffs to a large number, such as 100 Å.

arg1 *Monte Carlo frequency*

Number of molecular or stochastic dynamics steps for every Monte Carlo step in mixed mode MDMC simulations. The result of a block of arg2 Monte Carlo steps is one or more translations or rotations of coordinates and velocities. Any resulting structure passing the Metropolis test is used for subsequent molecular dynamics or stochastic dynamics simulation and accordingly modifies the trajectory.

0 Default value of arg1 is 1 for pure MDMC and 10 if importance sampling (IMPS) is being done.

<0 No randomization of internal degrees of freedom. This option is available only for MC(JBW/SD) simulations where all the randomization can be effectively done via the SD part.

>0 This argument may be varied to achieve the desired 100:1 ratio of stochastic dynamics steps to conformational interconversions when IMPS is used.

arg2 *Number of internal coordinates varied at each MC step*

Each pair of atoms in a TORS command corresponds to one internal coordinate degree of freedom. Each molecule moved by a MOLS command generates one (if only translation or only rotation being done) or two (if both translation and rotation limits given) internal coordinate degrees of freedom.

This argument is overridden by args 1 and 2 of the MCNV command. In addition, MCSD may dynamically override any user specification by using a built-in dynamic method of specifying limits on degrees of freedom varied. The adaptive mechanism may, however, be turned off by specifying DEBG 103.

0 Default: 1.

>0 Number of internal coordinates to be varied at each step.

<0 In JBW/MC (IMPS) do not randomize JBW step.

arg5 *MC acceptance ratio*

Desired ratio of accepted to total Monte Carlo steps. The program, during execution, adjusts the limits given in TORS and MOLS commands to attempt to achieve this ratio. Default: use limits as given in TORS/MOLS commands without adjust-

ment. We recommend using the default behavior, and making manual adjustments to TORS and MOLS if acceptance is too high or low.

MDVE — Molecular Dynamics VElocity file

This command directs MacroModel to write or read a velocity file to or from disk. The file has double precision x,y,z velocities for each atom. The filename is *out-filename*.vel when writing a velocity file and *in-filename*.vel when reading one. This option is useful for listing the velocities, for saving the velocities to continue a molecular dynamics run at some later time and as a mechanism by which externally generated velocities may be used as molecular dynamics starting points. Velocity files are stored as formatted disk files. Each line contains x,y,z velocities (meters/sec) for each substructure atom. The Fortran format for each line in the velocity data in the file is (3D20.12).

A velocity file read (arg1=1) must come after READ or SUBS commands and before any MDYN command. A velocity file write (arg1=0) must come after the MDYN command.

MDYN does not write final structures to the output file unless the WRIT command follows it. We generally save both final velocities and structures at the end of a molecular dynamics run.

arg 1 *Read/write a velocity file*

0 Write file.

1 Read file.

MDFR — Molecular Dynamics update FRequency

Sets the frequency in picoseconds for updates of the nonbonded array.

For molecules where large movements of groups are expected, the nonbonded array should be periodically updated (e.g., every 0.2-0.5 ps). A better alternative is to use very long cutoffs (using EXNB command) and a force field like OPLS, which does not have explicit hydrogen bonding potentials. The more frequently the nonbonded array is updated, the slower the dynamics will run.

arg5 *Update interval (fs) (default: do no updates)*

MDRE — Molecular Dynamics REset

Reset or reinitialize all molecular dynamics variables, set initial velocities to zero.

IMPS — perform IMPortance Sampling

Note: IMPS affects the course of MCLO and MCSD simulations. For the full description of possible combinations and what they do, see the *MacroModel Technical Manual*.

This command activates the importance sampling option. Use of this command implies that the input file contains the output from a conformational search. When this command is encountered, all the structures in the input file are read and the values of the specified torsions and bond angles calculated for each conformation. These are used later in the importance sampling. The energy and number of times the structure was found in the conformational search are also read from the input file. These quantities may be used to weight the importance sampling (this last option is not recommended, see below).

Importance sampling also activates calculation of conformational populations which can be used to obtain relative free energies (see MDSA command).

Importance sampling can be used either as a replacement or in conjunction with Metropolis Monte Carlo (MMC) in both Monte Carlo and mixed mode simulations. The method is described in detail in the technical manual and examples of command files are provided in Chapter 10 of the *MacroModel User Manual*.

arg1 *Frequency of performing IMPS steps*

0 Perform an IMPS step each time a MMC step is requested.

n Perform IMPS step every *n* times a MMC step is requested and a Metropolis Monte Carlo step the rest of the time.

<0 Do not perform any IMPS steps. This option can be used to tabulate statistics of visiting conformations in an input list during an ordinary MDYN or MCSD run. For example, using IMPS with a negative first argument results in a standard MCSD run; if, in addition, the first argument of MCSD is negative, ordinary stochastic dynamics is run.

arg2 *Control of trial conformation*

0 All trial conformations including the current one are equally probable; that is, a jump may be attempted to the current conformation.

1 All trial conformations except the current one are equally probable; jumps are never attempted to the current conformation.

2 The probability of choosing a conformation is weighted according to the number of times it was found in the conformational search. This option is not recommended.

arg7 *Time_Limit (ps for dynamics, step for MC)*

If the simulation spends more than this amount of time (or number of steps) consecutively away from known conformations, the program prints the last structure to the output file and stops.

0 Default: 50

>0 Number of ps or steps.

<0 Multiply by 100000, then interpret as number of ps or steps.

ITOR — Importance-sampling a TORsion angle

This command is now obsolete.

IMPO — IMPortance sampling Options

This command controls options available for the IMPS procedure.

At the beginning of an IMPS run, the program checks the internal coordinates by making all possible n^2 conformational interconversions between the n input conformers. The program produces an n by n matrix whose (i,j) element gives the difference between the energy of conformer j obtained from conformer i and its actual energy as read from the input file. Ideally, these energetic differences should all be zero, but in practice they are nonzero, since the interconversion is accomplished by means of torsional and bond-angle transformations alone. Energetic differences larger than a predetermined threshold lead to the termination of the run, unless an override is specified. The n^2 structures obtained by these interconversions may be written to the output file. Superimposing high energy structures (those for which the energy difference exceeds the threshold) on their original counterparts may help in identifying the cause of large energy differences. In particular, the deletion of a crucial torsion or bond angle leads to these symptoms, and the procedure just described allows such missing degrees of freedom to be identified.

Arguments 1, 2, 3, and 5 control the process described above. Argument 6 controls the criterion for declaring a conformation encountered during a simulation to be an instance of one of the IMPS input conformations, and Argument 7 controls the action to be taken when an importance-sampling step is scheduled, but the starting conformation does not correspond to any IMPS input conformation.

arg1 *Control the printing of the energy difference matrix*

0 Print the energy differences matrix to the log file if any of its elements exceeds the energy threshold (arg5). Such elements are marked with an exclamation point. This is the default.

1 Always print the energy difference matrix to the log file.

arg2 *Control action if energy differences exceed threshold*

0 Abort the simulation.

1 Continue to carry out the simulation. the run. This is recommended when (i) the energy differences do not significantly exceed the threshold, or (ii) the conformational pairs that exceed the threshold are high energy conformers which are not expected to be significantly populated during the simulation.

arg3 *Control writing structures to output file*

<0 Write no structures to the output file.

0 Write only high energy structures (i.e., those for which the aforementioned energy difference exceeds the threshold) to the output file.

>0 Write all interconverted structures to the output file.

arg4 *Frequency of* `.ino` *output*

This argument is most useful for debugging the `IMPS` facility.

0 No output

n>0 Write into *outfile*`.ino` the index of the input structure that most closely matches the current simulated structure every *n* MC steps or SD time steps. When the input structure index is followed by an asterisk (*) in the file, this means that the calculated RMS difference between the simulated structure and the indexed input structure exceeded the threshold defined in arg6. This can happen only for torsional RMS calculation.

arg5 *Interconversion energy threshold*

Maximum allowed energy difference between a conformer's energy as provided in its *filename*`.mae` title line and its energy obtained by `IMPS` interconversion from another structure. Any energetic difference larger than arg5 causes the energy-difference matrix to be written to the log file, or the offending structures to be written to the output structure file, or the run to be aborted, or any of these, depending on the values of arguments 1, 2, and 3.

0 10 kJ.

arg6 *Interconformational comparison control*

At each IMPS step, the program assigns the currently simulated structure to one of the conformers provided in the input file—the one it is most similar to—provided this similarity is strong enough. This argument defines the meaning of "similarity."

>0 Uses the torsions defined in TCMP commands to calculate torsional RMS difference. The actual value of this argument is not significant, provided it is positive.

<0 The atoms listed in COMP commands are used to compute a Cartesian RMS difference. JBW steps are performed even if the RMS difference is larger than the value defined here. However, conformational populations do depend on this value and will not be updated if the computed RMS difference exceeds it.

ITBS — Improper Torsion energy BiaSing

This command is used to bias the energy of a conformational family with a proper or improper torsion (also with a proper torsion) in a specific range by a constant amount. If, for example, the steric binding energies of conformations of L-family ligands are much lower than those of the D-family ligands, then attempts to calculate the enantioselectivity by jumping between the L and D families is likely to fail, because most L → D jumps will be rejected. In such cases, it is possible to artificially stabilize the D family by subtracting from the steric energies of its members a constant amount. This number must then be added back to the free-energy difference at the end of the simulation. If, for example, an energy value of 1.5 kcal/mol is subtracted from the members of the D family during the run, leading to a D/L free-energy difference of 0.5 kcal/mol in favor of the latter, then the real free-energy difference is 0.5 + 1.5 = 2 kcal/mol.

arg1-4 *The numbers of atoms defining the improper (or proper) torsion.*

arg5 *Minimum value for the improper torsion.*

arg6 *Maximum value for the improper torsion.*

arg7 *Value for energy bias in kJ/mol.*

IMCC — IMportance sampling Chirality Check

This command is used to define chiral centers to be checked when assigning the simulated structure to one of the input structures. The current structure will not be assigned to an input structure if the chirality of one of its centers is different from that of the input structure. This command must come before the IMPS command.

arg1-4 *Atom numbers of chiral centers*

TCMP — Torsional CoMParison

Used during importance sampling to define the torsions for torsional RMS calculation and for the torsional check during the assignment of the simulated structure to one of the input structures. Can be used either with torsional RMS calculations (both functions) or with Cartesian RMS calculations (second function only).

arg1-4 *Atom numbers defining the torsional angle*

ZMAT — Z-MATrix

Defines the internal degrees of freedom (Z-matrix variables) to be sampled during all degrees of freedom MMC or MC(JBW) runs (also used in MCSD and MC(JBW)/SD). In a Z-matrix, each atom (except the first three specified) is defined with a bond length, bond angle, and torsional angle with respect to atoms previously defined. The first atom is defined with respect to itself, the second by a bond length to the first, and the third by a bond length to the second and a bond angle to the first.

Also defined in this command are the ranges of change/randomization for each degree of freedom. Typical ranges are 0-0.1 Å for bond lengths and 5 for bond angles. In course of an MMC simulation, backbone torsions should be changed by large ranges to allow for a complete sampling of the entire potential energy surface. With the above definition of Z-matrix, torsions to branched atoms should be changed by a small amount (0-5) to avoid distorted bond angles. For MC(JBW) and MC(JBW)/SD simulations, typical randomization ranges are 0-0.1A for bond angles and 0-5 for bond and torsion angles. Large randomizations reduce the acceptance rate and the number of conformational interconversions, and have the effect of reducing the JBW procedure to an MMC one.

arg1-4 *Atom numbers*

Specify two for bond lengths, three for bond angles, four for torsions.

arg5 *Minimum change/randomization range*

No default values are supplied; zero means zero.

arg6 *Maximum change/randomization range*

No default values are supplied; zero means zero.

arg8 *Atom-connectivity sanity check*

Check to make sure the atoms specified in arg1-4 are connected.

0 Default: Perform this test.

1 Do not perform this test (required for improper torsions).

4.12 Free-Energy Perturbation

See Chapter 15 of the *MacroModel User Manual* for an extended discussion, including a descriptive example.

FEGA — Free Energy mutating Group Atom

This command allows explicit addition to the list of atoms whose parameters change during a free-energy perturbation simulation. It is rarely used, since ordinarily the free-energy perturbation group atoms are taken to be those atoms that change atom type between the starting and final structure, together with any atoms bonded to them. This command could be used to add to this list, in the event, for example, that the parameters of an atom beta to an atom undergoing mutation changed greatly due to the alteration of the mutating atom.

Any FEGA commands must precede all other free-energy perturbation commands.

arg1-4 *Atom numbers*

<0 Add all atoms in the molecule containing atom |arg| to the mutating atom group.

>0 Add this atom to the mutating atom group.

FEIA — Free Energy Interaction Array generation

Generate the two interaction arrays for averaging. It is assumed that the input file contains two structure files which represent the starting and end points for the mutation. It is further assumed that these structures have the same number and ordering of all atoms. Dummy atoms (MacroModel type Du) should be used as counterparts for atoms which disappear or appear during the simulation.

All constraints (FXAT, FXDI, etc.) must be specified before FEIA if the constraints are to be active during the free-energy perturbation run.

FEAV — Free Energy AVeraging

This command, together with FESA, ordinarily appears in a BGIN/END loop that sets up the simulation for each window in turn. The first time this command is encountered, the interaction array is generated with the value of lambda given by arg5. You would specify a nonzero value for arg5 in order to simulate some individual window, only, or to start a series of windows with some nonzero value of lambda. This might be useful, for example, if the computer crashed part way through an earlier simulation attempt. It is also used in network-distributed processing to instruct a given processor to run a simulation over only a single window.

arg1 *Averaging protocol*

0 Altered parts of interaction arrays will be averaged.

1 Intragroup interactions (see FEGA command) will be zeroed in the final interaction array (used primarily for testing).

arg5 *Coupling factor (λ) for the current window*

This is the fraction of structure 2 (the mutation end point, the second structure in the input file) mixed into the molecular representation in the current window.

FESA — Free Energy SAmpling

Sampling protocol for a particular perturbation window. A maximum of 100 FESA commands (defining up to 100 free-energy perturbation windows) can be used. FESA is normally used within a BGIN/END loop to carry out the entire mutation. Arg7 and arg8 can be used to allow automatic extension of simulation time in a given window to improve convergence.

arg2 *Maximum allowed number of step doublings*

Simulation time (MDYN, MCSD) or number of steps (Monte Carlo) can be successively doubled to achieve user-defined convergence criteria. This argument allows control over this. See also arg7, arg8.

0 2, if arg7 or arg8 are nonzero.

arg5 *Control of double-wide sampling—left half*

Fraction of structure 1 for the left half of the double-wide sample. If arg5 is equal to the current window's value of λ, then no left sampling is done.

arg6 *Control of double-wide sampling—right half*

Fraction of structure 1 for the right half of the double-wide sample. If arg6 is equal to the current window's value of λ, then no right sampling is done.

arg7 *Minimum allowed standard deviation (kJ/mol)*

If the standard deviation of the free-energy difference for this window exceeds this value at end of the simulation, the simulation is doubled in length, in an attempt to improve the statistical uncertainty in the calculated free energy. A maximum of arg2 such doublings are allowed. By default, this option is off.

arg 8 *Maximum allowed difference in free energies (kJ/mol)*

If the free-energy difference between the current and the previous window exceeds this amount at the end of the simulation, the simulation will be doubled in length in an attempt to improve convergence. A maximum of arg2 such extensions are allowed. By default, this option is inactive.

FESU — Free Energy SUmmary

List summary of accumulated free-energy perturbation results for all sampling windows to `.log` file.

arg1 *Energy units for output.*

0 Use kJ/mol.

1 Use kcal/mol.

4.13 Monte Carlo Simulation

MCLO — Monte CarLO simulation

Carry out classical internal coordinate Metropolis Monte Carlo simulation for sampling of phase space. Use MDYN, MCSD or MCLO commands to identify internal degrees of freedom to be varied. RCA4 commands are allowed, but a low acceptance ratio can be expected unless IMPS is being used.

The MDSA command provides control over the extent of output in MCLO as well as MDYN procedures.

arg1 *Extent of listing*

0 No listing.

1 Listing to `.mmo` file.

arg2 *Number of internal coordinates varied at each MC step*

0 Each MDYN, MCSD or MCLO command corresponds to one internal coordinate degree of freedom. Each molecule moved by a MOLS command generates one (if only translation or only rotation is being done) or two (if both translation and rotation limits are given) internal coordinate degrees of freedom.

This argument is now overridden by args 1 and 2 of the MCNV command. In addition, be aware that MCLO may dynamically override any user specification by using a

built-in dynamic method of specifying limits on degrees of freedom varied. The adaptive mechanism may be turned off with DEBG 103.

arg3 *Number of MC steps in simulation*

>0 Perform this number of steps.

<0 Perform 100,000 times |arg1| steps. This facilitates specifying a large number of steps. (Note that prior to MacroModel 6.5, the multiplier was 100.)

0 Note: Not a valid argument.

arg4 *MC restart frequency*

Every arg4 steps, the original starting geometry is retrieved and the simulation is continued with it.

arg5 *Desired MC acceptance ratio*

Desired ratio of steps which find an acceptable structure to total steps taken. Limits given in MDYN, MCSD or MCLO commands are adjusted to obtain desired acceptance ratio.

0 Use MDYN, MCSD or MCLO values without adjustment.

arg7 *Temperature (K)*

Must be greater than zero.

4.14 Docking

MBAE — Multi-ligand Bimolecular Association with Energetics: Embrace

MBAE invokes an algorithm in which MacroModel performs calculations on a number of pre-positioned ligands with a single receptor. The input structure file must must be a multiple structure file consisting of a receptor followed by each pre-positioned ligand in turn. MBAE works in two modes, interaction mode and energy difference mode. In the former the calculation examines the interaction between the receptor and the ligand, while in the latter the calculation is performed on the receptor, the ligand, and then the complex.

Values calculated for the energy difference mode are given by the equation:

$$\Delta E = E_{\text{complex}} - E_{\text{ligand}} - E_{\text{protein}}$$

This command is implemented in conjunction with minimization and conformational searches. Only LMCS, MCMM or mixed LMCS/MCMM searches are supported for conformational search MBAE. Please note that MBAE conformational searches may require the purchase of an additional license.

Energy difference mode is the only mode permitted for conformational search MBAE. The values of E_{protein} and E_{ligand} are the lowest energies found for any conformer of the protein and ligand respectively. ΔE is calculated for each complex conformation saved in the output structure file using the energy of the current complex conformation for E_{complex}. The number of output structures can be controlled independently of the number of steps performed in the search, via arg6 of MCOP. MBAE can also be used in conjunction with COPY/ALGN and MCOP arg7 to pre-position ligands for MBAE conformational searches based on a crystal structure of a related complex. The .com file for MBAE conformational searches is complex and specific, and consulting Section 14.2 of the *MacroModel User Manual* for MBAE searches is recommended.

Both minimization and conformational search MBAE calculations may be distributed across multiple processors. If such calculations are carried out in energy difference mode then the calculation on the receptor is carried out by the parent process prior to distributing the calculations for the ligands amongst the processors indicated. Please see the NPRC command for more information. Please note that arg3=2 is not supported for distributed MBAE calculations.

A substructure file should be used to specify substructures and various fixed/frozen constraints within the receptor.

arg1 *Selection of Association Energy mode*

−1 Turn MBAE off.

0 Use Interaction Energy mode.

The interaction energy between the ligand and the receptor is calculated using the ASET mechanism. For non-covalently bonded receptor/ligand pairs, this corresponds to the nonbonded energy. While a number of sets may be used, the only project properties recorded in the output structure file are those for the interactions between sets 1 and 2.

Note: For jobs set up in Maestro, ELST arg1 is set to −1 in the .com file when MBAE is run in Interaction Mode (i.e., when MBAE arg1 is set to 0).

1 Use Energy Difference mode.

The calculation is performed on the receptor first, then it is carried out on the ligand, and finally it is performed on the complex. All calculations start from the structures provided in the input structure file.

arg2 *Type of calculation being performed*

0 Minimization (`MINI`) calculations are performed.

1 Conformational search calculations are performed.

arg3 *Structural output*

0 Only the resulting complex structures are recorded.

1 Only the resulting ligand structures extracted from the resulting complex structure are recorded.

2 For Interaction Mode calculations this is the same as arg3 = 0.

For Energy Difference calculations, structures are recorded in the following order:

Processed Protein structure (no ligand).

For each ligand:

The processed ligand structure (isolated, no protein).

The processed complex structure.

For arg3 = 2 the `MBAE` project properties are recorded with the complex structure only.

Arg3 = 2 is not supported for distributed calculations.

arg4 *DEBG 510 (Do not enforce properties dependencies)*

0 Turns on `DEBG` 510 (default).

1 Turns off `DEBG` 510.

4.15 Miscellaneous Commands and Debugging

DEBG — DEBuGging

Turn on program debug switch (numbered 1-1100). Use as many of these commands as necessary to turn on as many of the switches as you want.

arg1-4 *Debug switches*

Values currently in use are listed below. "GV" stands for "Greater Verbosity." Flags described with this abbreviation affect only the level of reporting; other flags affect program action.

1 GV, all commands

2 Don't line-buffer `.log` file

3 Always write full (never compressed) output files

5 GV, dynamic allocation

6 Always write old-style (mmio) files

7 Always write new-style (m2io) files

8 GV, correspondence between mainCT and sbstrCT numbering

9 GV, handling of mmio/m2io files

10 GV, `RWND` command

11 Get formal charge from `.fld` instead of `atom.typ`

12 Don't set qq product to zero, `VDWB` nonbondeds

13 GV, `LIGB`, and `VDWB` bonds

14 GV, `SUBS`, `FXAT`, `FXDI`, `FXBA`, `FXTA` command handlers

15 GV, reading charges from ffld substructures

16 GV, VDW offset from ffld

17 Suppress elimination of constrained-atom mutual interactions

18 GV, charge delocalization

19 GV, number of matches, each ffld substructure in mol

20 GV, name of each ffld substr matched in mol, and atom #s

21 GV, more details on how each ffld substr matched mol

22 GV, write problematic structure to output structure file if `MINI` fails

23 GV, write `MINI` structure to output structure file at each `MINI` display interval. Incompatible with `DRIV`.

25 Don't refuse to minimize extremely strained structures

27 GV, TNCG, FMNR `MINI` methods

28 Suppress elimination of stretching and bending interactions in which some of the atoms are fixed.

30 GV, `MINI` line search

31 GV, `MINI` gradient to log file, each iteration

32 Don't use constant derivatives in line searching

33 Don't separate close pairs during interaction generation

34 GV, tests for distorted sp3 atom and for chirality

35 Do write output `.sbc` file

36 GV, do not limit the number of messages from close separation problems

37 Note that if separation of close pairs fails in a conformational search reject the structure and continue the search

38 Skip tests for distorted sp^3 atoms.

40 GV, early in force-field (eqn. reading, etc.)

41 GV, BMFF IPC layer (client and server side)

42 GV, BMFF server

43 GV, BMFF-related stuff within MacroModel

44 GV, applying mmlewis and obtaining OPLS_2001 or OPLS_2005 parameters

45 [NEW] Do not delete dummy atoms for OPLS_2008. Dummy atoms are correctly handled in OPLS_2008.

46 Don't use fast vdw and hbd BMFF processing

49 Don't skip over structures that do not have all force field and solvation parameters. Stop when such a structure is encountered.

50 GV, parsing of `atom.typ` file

51 LV, parameter assignment

55 [NEW] Don't compare the connectivity of the input structure with the previous structure, but treat it as chemically different from the previous structure.

56 Don't eliminate torsional interactions with V's of 0

57 Always consider input structures to be distinct CT's, not conformations. This implies full interaction generation for each structure read in.

58 GV, when which classes of params are updated (Bonded, VDW, Geom-dependent, solvation)

59 GV, geom-dependent params & updating

60 GV, `.fld` ALT and SEL processing

61 GV, stretch interactions

62 GV, bend interactions

63 GV, torsional interactions

64 GV, nonbonded interactions

65 GV, bend-bend interactions

68 GV, `.fld` atom-type equivalencing

69 GV, `.fld` line-numbering & assembly of interactions; to `.mmo`

70 GV, fixed atom interactions

71 GV, parameters that change during FEP

72 GV, atoms whose surface areas change during FEP

73 Don't eliminate str/bend involving dummy atoms or zero-order bonds in FEP accumulation

74 Exclude all str/bend terms from FEP accumulation

78 Calculate normal modes only for SUBS atoms in LMOD and LLMOD calculations

79 ATEQ, also consider noncyclic permutations of the ATEQ atoms

81 GV, ATEQ, during csearch or mult MINI test for uniqueness

82 GV, Comparing structures for uniqueness

83 GV, MOLS and TORS moving-atom sets

85 Use DRVPOL_OLD instead of DRVPOL; DRVPOL is faster for 1st derivs but has bad 2nd derivs. We use DRVPOL_OLD when we need 2nd derivs or when DEBG 85 is specified

86 Use current rather than ideal str and bend distances in computing constant part of analytical GB radius

87 GV, atom-wise solvation data to .mmo even if E(atom)=0

88 GV, read .slv solvation file and match its atoms to mol

89 Don't unite all-atom sp3 CHn groups for solvation

90 Include explicit hydrogens in GB-radius calculation

91 Use old (Hasel et al.) method for analytical surfaces

92 GV, low-mode search (LMCS or LMC2)

93 Turn off surface-area 3-body function

94 GV, saddle point search. Also, write intermediate structures to output structure file.

95 Use all-atom representation in surface calcs (assuming molecule actually has H's and/or lone pairs)

96 Ignore long range shells in numerical Born rad. calc

97 GV, solvation energy calc

98 GV, parameters for overlap array, solvation calc

99 GV, analytical and numerical GB radii and Gpol values for fixed atoms

100 Set negative areas to zero in LCPO surface calculation

101 GV, MD or MC monitor of PE avg, sd, skew and kurtosis

102 GV, timings, MD and `MINI`

103 Turn off adaptive mechanism for number of Monte Carlo degrees of freedom during `MCSD`, `MCLO`, `IMPS`

104 After MD "Abort," continue with remainder of `.com` file

105 Do not do LCPO Buried Atom Elimination (BAE)

106 GV, SHAKE

107 Do not do neighbor-list reduction (NLR) on LCPO overlap array

108 GV, write LCPO `ELST 3` results to *filename*`.lcp` as well as `.mmo`. If `DEBG 91` is in effect, still write the file, which includes the numerical atomic surfaces, but with LCPO sums set to 0.

111 GV, general dynamics & MC

112 GV, write every 200th non-matching `IMPS` structure to output structure file

121 GV, `BDCO` pairlist generation

126 GV, `BDCO` total charge product

127 GV, `BDCO` pairwise charge product assignment

128 GV, more `BDCO` pairwise charge product assignment

129 GV, force field substructure explicit partial charge processing

130 GV, `CHGF` input structure file explicit partial charge to delocalized formal charge + bond charge increment decomposition

131 GV, details of lone-pair requirements for current ffld, mol

141 GV, structures accepted, etc., MC csearch

150 During dihedral driving start each incremental minimization from the initial structure as read from the *filename*`.mae` file

152 GV, interaction array changes from nonbonded update, `MINI`

169 GV, torsional angles stored by torsional memory

178 Use MacroModel-supplied random-number function

179 GV, print seed to log file each time random-number generator called

180 GV, movements used to generate new Monte Carlo structures

181 GV, internal coords used in tors MC and `MCSD` simulations

182 GV, internal coords, final limits after tors MC run

183 GV, internal coord changes at each tors MC acceptance

184 In MC, calculate total E, not just MC-variable E components

185 Allow JBW in substructures—for internal use only

186 Don't reorient each output structure to best superimpose on input

187 GV, For `IMPS`, write each input structure to output, after zmat superposition of the first three atoms onto those of the first structure. Requires user specification of `DEBG 186`.

188 Skip problematic ring closure rotations. This may result in chirality switches if `CHIG` commands are not applied to the two central atoms in the ring closure.

191 Don't record OPLSAA formal charges or bond orders in the output structure file. Use the ones from the input structure file.

200 GV, `CGEN` searches

210 GV, mass-weighted force-constants, `RRHO` command

211 Allow `RRHO`, `MTST` on structures not necessarily minimized

222 GV, write starting structure for each `MINI` to output structure file

230 GV, `COPY` opcode

235 GV, `ALGN` opcode, general

236 GV, `ALGN` opcode, center of mass operations

237 GV, `ALGN` opcode, principal axis operations

251 GV, "torsional memory" during conformation search

252 GV, conformational search of ring structures

333 GV, network-distributed processing using the NPRC command

358 GV, geometrical pre-optimization of ring structures

360 Turn off use-directed csearch initialization from *filename*.mae file

370 Enable MOLS for molecules containing fixed atoms

400 GV, importance sampling

500 GV, source CT number tracking for Maestro project facility properties

510 Do not enforce properties dependencies (i.e., do not clear properties when the structure changes) In Embrace calculations it is turned on by default. Use arg4 of MBAE to turn it off.

511 GV, the process for recording properties

512 Do not add dependencies to MacroModel Properties that lack them.

513 Record energy components in the output structure file. This flag is on by default; use DEBG 514 to turn it off.

514 Do not record energy components in output structure file. Turns DEBG 513 off.

520 Basic reporting for Automatic setup (AUTO)

521 GV, reporting for Automatic setup (AUTO) and turn on DEBG 520

530 Record the structure for each molecule in both solvents for LOGP calculations in the output structure file

531 GV, for LOGP calculations

550 GV, MBAE

555 GV, LOOP

570 Don't turn on MMSYM_IN_PLACE when a fixed or frozen atom is added. The use of DEBG 570 is not recommended because mmsym comparisons which are not done in place translate and rotate the system without taking into account fixed or frozen atom positioning, resulting in distorted structures.

601 Turn off writing to output structure file unless explicit WRIT is used

602 If old-style titles are detected in a Maestro-formatted input structure file, do not update them.

720 GV, 1st and second derivs of Wilson angle OOP terms

725 Turn off constant long-range nonbonded derivs

726 Turn off constant long-range polarization solvation terms

727 Turn off constant long-range surface-area solvation terms

800 Don't regenerate solvation overlap array after MC accept

820 Set all atomic weights to 12 for dynamics, inc. FEP

825 Don't correct tiny denom. in analytical solvation derivs

830 Don't correct Gpol,i' for nonbonded pairs left off the nonbonded pairlist

832 Don't correct Gpol,i' for F-F nonbonded pairs which are farther apart than the nonbonded cutoff

F = (a) fixed without flat bottom well or (b) frozen

835 GV, final Gpol,i' values for all atoms

836 GV, generalized Born CCF (Close Contact Function). Available only with single precision energies)

899 GV, FlexLM licensing general

900 GV, print RCS versions of source files used in compilation

901 GV, detailed tables of COMMON memory utilization

902 GV, table of atom info (regurgitation of `atom.typ` info)

903 GV, table of atomic masses by atom number

920 Low-mode search (`LMCS` or `LMC2`), write perturbed structures in output structure file

930 Do not check for `.stp`, `.upt`, `.slp` files (speeds response, especially on NFS-mounted filesystems, but loses the functionality of these checks). This behavior is on by default. To enable checking for `.stp`, `.upt`, `.slp` files, use `DEBG 931`.

931 Turns DEBG 930 off, thus turning on checking for .stp, .upt, .slp files. Note that these files must be in the directory from which the MacroModel job is being run. It may be useful to use DEBG 931 with the -LOCAL command-line option to MacroModel, which keeps job files in the current directory.

940 GV, for network-distributed MacroModel jobs using the NPRC command, save files written by slave processes. You must use the -LOCAL option to the bmin launch script when using this debug option.

950 GV, on IBM SP2, save file written by parallel threads

960 Turn off reporting of format problems in the .com file

961 Cause MacroModel to stop if a format problem is found in the .com file. This overrides DEBG 960 and forces reporting of format problems.

975 Do not use atomic number information from the input file. Instead, obtain the atomic number information from the atom type.

999 GV, ARPK

1000 GV, give CPU timings in energetic routines. Now gives more detailed and informative output.

1001 GV, MINTA free-energy calculation.

DUMP — DUMP connection table

List the connection table to the log file (used primarily for testing purposes).

GEOM — obtain GEOMetric information about the molecule

The GEOM command allows the user to obtain geometric information about the molecule from the MacroModel command file. Bond lengths, bond angles, and dihedral angles are written as properties to the structural output file.

arg1 *Atom 1*

If arg1 is nonzero and arg2 is zero, then GEOM prints out the x, y, and z coordinates of the atom given by arg1.

arg2 *Atom 2*

If arg1 and arg2 are nonzero and arg3 and arg5 are zero, GEOM prints the distance between the atoms specified by arg1 and arg2.

arg3 *Atom 3*

If arg1, arg2, and arg3 are nonzero and arg4 is zero, `GEOM` prints the angle between the atoms specified in arg1, arg2, and arg3. If arg5 is nonzero, it is ignored.

arg4 *Atom 4*

If arg1, arg2, arg3, and arg4 are nonzero, `GEOM` prints the dihedral angle between the atoms specified in arg1, arg2, arg3, and arg4. If arg5 is nonzero, it is ignored.

arg5 *Spin-spin coupling constant*

If arg5 > 0, `GEOM` prints the spin-spin coupling constant *J* between the atoms specified in arg1 and arg2, subject to the following conditions:

- The atoms given by arg1 and arg2 are hydrogen atoms
- The hydrogen atoms are connected through two carbon atoms
- The values of arg3 and arg4 are zero.

If either of the first two conditions is not met, a warning is printed and the calculation continues without calculating the coupling constant.

`TIME` — report cpu TIME (user+system)

The `TIME` command reports CPU time (user+system) since the previous invocation of `TIME`, or (on the first invocation) since program start. See also `DEBG 1000` for more detailed and informative output on task timings.

`JWRT` — Journal WRiTe

The `JRED` and `JWRT` commands are provided in order to use MacroModel as a force-field server in connection with procedures being carried out by other processes. `JWRT` writes out a user-specifiable combination of coordinates, energy, gradient, and Hessian in binary form that the co-process can read. `JRED` reads in coordinates in a similar format that have been written out by the co-process. Thus, the co-process can provide coordinates to MacroModel, instruct it to compute the energetic terms, read back those energetic terms, manipulate the molecule, and complete the cycle. The way this facility is provided now, MacroModel has to be launched anew for each new calculation (for example, by a `system()` call from the client process); however, we may later enhance this facility to make it more fully interactive.

arg1 *Write coordinates*

If arg1 is nonzero, write out the coordinates.

arg2 *Write energy*

If arg2 is nonzero, write out the energy.

arg3 *Write gradient*

If arg3 is nonzero, write out the gradient.

arg4 *Write Hessian*

If arg4 is nonzero, write out the Hessian.

The filename used is *filename*`.jwr`, where *filename* is the prefix of the input filename. The file is opened with FORM=UNFORMATTED, but is sequential. All variables are written in double precision. Thus, if all four args are nonzero, and if there are N atoms in the part of the system being simulated (the whole system, unless `SUBS` is used with `FXAT` and all atoms are not covered by these commands), the `.jwr` file will contain $3N$ coordinates, in sequence x(1), y(1), z(1), x(2), ..., z(N), followed by one double precision value for the energy, followed by $3N$ gradient components, followed by $(3N)^2$ Hessian elements, in order x(1)x(1), y(1)x(1), z(1)x(1), x(2)x(1), ..., y(N)z(N), z(N)z(N).

`JRED` — Journal REaD

arg1 *Read coordinates*

If arg1 is nonzero, read the coordinates.

arg2 *Read energy*

If arg2 is nonzero, read the energy.

arg3 *Read gradient*

If arg3 is nonzero, read the gradient.

arg4 *Read Hessian*

If arg4 is nonzero, read the Hessian.

arg5 *Print values read*

If arg5 is nonzero, whatever is read will be printed to the `.log` file. However, only the coordinates are used by subsequent computations in the program; the energy, gradient, and Hessian are simply read and printed, if requested.

JRED works as follows. It expects to read a file called *filename*.jrd, where *filename* is the stem of the input file name. For the purpose of debugging the JWRT command, args1-4 cause JRED to read the coordinates, the energy, the gradient, and the Hessian, respectively, if nonzero. If arg5 is nonzero, whatever is read will be printed to the .log file.

NPRC — Number of PRoCessors (distributed MacroModel calculations)

Distribute the MacroModel job over a number of different hosts. The name of the hosts to use are taken from the file schrodinger.hosts, which must be in the $SCHRODINGER directory or the user's working directory. The NPRC command can be used with the following kinds of calculations:

- Non-serial conformational searches
- Serial conformational searches
- MULT minimizations
- Free-energy perturbation calculations (FEAV, FESA)
- Embrace (MBAE) minimization and conformational search calculations

The types of non-serial conformational searches supported are: MCMM, LMCS, LMC2, mixed LMCS/MCMM, and mixed LMC2/MCMM. The types of serial searches supported are MCMM, LMCS, and mixed LMCS/MCMM. The AUTO command is required for serial MCMM and serial mixed LMCS/MCMM jobs, and is recommended for serial LMCS jobs. The procedure is split into a number of different searches and run on the hosts as they become available. In an MCMM procedure, a different SEED value is used on each host.

For MBAE energy difference calculations the initial receptor computations are conducted by the parent process prior to distributing the calculations for the ligands amongst the child processes. Because of the complexity of MBAE computations users are encouraged to consult the description of the MBAE opcode in this manual and the examples of MBAE calculations in Section 14.4 of the *MacroModel User Manual*.

See Section 2.3 on page 19 for a description of internally-distributed MacroModel calculations which use NPRC and see Chapter 6 of the *Installation Guide* for information on setting computers and accounts to permit distributed MacroModel calculations.

If a requested host becomes unavailable during a run, its status is checked periodically. If the host later becomes available, the process starts using it again. Exception: hosts unavailable at startup time are removed from the list of allowed hosts and never re-used. This is because a host that was unavailable at the start of the job cannot have been checked for energetic consistency (see arg4 below), and using it without this test having been performed is potentially dangerous.

DEBG 333 gives greater verbosity; DEBG 940 saves, rather than discards, the output files of the intermediate jobs.

arg1 *Number of hosts to check for availability*

arg1 specifies how many hosts in the `schrodinger.hosts` file (see Section 2.2 on page 17), will be checked for availability for a specific distributed run. If any hosts are not available, they are removed from the list of hosts to be used for the specified job. Each host that is checked but unavailable decreases the number of hosts over which the job will be distributed. Only hosts read and available will be used.

For example, if the `schrodinger.hosts` file lists ten hosts, and arg1 is set to 5, then the first five hosts listed in the `scmchrodinger.hosts` file will be checked for availability. If two of the first five hosts are unavailable, the job will be distributed over three hosts.

arg2 *Job size*

For non-serial searches arg2 specifies the number of steps from the search that should be carried out by each job. The total number of steps is the number given by arg1 of the search command (i.e. arg1 of `MCMM`, `LMCS`, or `LMC2`) and the total number of jobs is roughly given by arg1 of the search command divided by arg2 of `NPRC`.

For serial calculations, such as minimizations of multiple input structures and serial searches this argument specifies the number of input structures given to each job for minimization or searching. In serial searches separate searches are conducted for each structure in the input structure file. Serial searches can be conducted for `MCMM`, `LMCS`, `LMC2`, mixed `LMCS/MCMM` and `SPMC` searches. The total number of jobs is given by the number of structures in the input structure file divided by arg2 of `NPRC`.

If the total number of jobs is significantly larger than some small multiple of the number of hosts specified by arg1 of `NPRC`, load balancing can take place by sending more jobs to the machines that finished their assigned jobs faster. The number of jobs should not be too large, as this would result in overhead associated with starting many jobs and organizing the results from many jobs.

0 For non-serial searches, set the number of steps per job to the total number of steps for the search divided by the number of processors (arg1).

For serial searches, set the number of structures to process per job to the number of structures in the input structure file divided by the number of processors.

In either case, this setting is sub-optimal since no load balancing can be carried out.

arg3 *Sleep time (sec)*

The time in seconds that the "master" MacroModel process will sleep before checking on the progress of the "slave" jobs.

0 60 seconds. This is a reasonable value.

arg4 *Energy-test control*

0 Perform no energy tests.

1 Perform energy tests; this is highly recommended.

Prior to starting the distributed job, an energy calculation on the starting structure is performed on each processor. The resulting energies are compared. If any value differs from that on the current processor by more than 0.001 kJ/mol or 0.1% (whichever is smaller), the user is notified and the job is terminated. This is to make sure that the version of MacroModel and the associated solvent and force-field files are the same on all hosts.

SPAT — SPecial Atom Treatment

This opcode is used to instruct MacroModel how to proceed when it encounters certain atom types.

arg1-4 *Atom types to be flagged*

Specify which atom types are flagged by MacroModel for special treatment. See Section C.3 on page 182 for a listing of MacroModel atom types.

arg5 *Action to be taken on flagged atom types*

1 Instructs MacroModel to stop when it encounters any of the atom types specified in args 1-4.

Multiple SPAT lines may be used, and arg5 may differ on each of them (it applies only to the atom types given in the same SPAT line).

4.16 ConfGen

ConfGen (CGEN opcode) is a very flexible and fast conformational searching technology for ligands. This section lists the opcodes associated with the ConfGen facility. Full details of the ConfGen facility, including the Maestro interface, are given in the *ConfGen User Manual*.

ConfGen is a separate module and requires additional licensing. The opcodes listed here are nonfunctional without such a license. See the *Installation Guide* for information on obtaining licenses.

CGEN — ConfGen

arg1 *Maximum number of structures to request from ConfGen*

arg2 *Maximum number of structures to retain while running*

0 The maximum number of conformations to retain while running is the value specified in arg1 or 10,000, whichever is greater

>0 This is the maximum number of conformations to retain while running

arg4 *Peripheral sampling option*

1 Rapid Sampling: only generate conformers in which at most one peripheral group is rotated away from its lowest internal energy conformation

2 Thorough Sampling: sample all combinations of rotations of peripheral groups

arg6 *Allowable interatomic approach distance*

Fraction of sum of van der Waals radii which is used as a closest atomic approach limit (default: 0.25)

arg8 *Verbose reporting on* CGEN *processing*

Related DEBG flag: 200.

CGOP — ConfGen OPtions

CGOP is one of a number of opcodes (CGOP, CGO2, CGO3, CGO4, CGO5, CGO6) that provide access to additional options for the ConfGen facility.

arg1 *Sampling symmetric terminal groups*

Terminal $-CH_3$, $-NH_2$ and $-NH_3^+$ groups are never sampled by ConfGen. Sampling the conformations of other types of symmetric terminal groups is sometimes not useful. The identification of such groups is complex but boils down to an atom with identical groups attached to it. These groups can be monoatomic or composed of atoms with only 2 or 3 hydrogen atoms bonded to them. Checks for rotational symmetry are also imposed (e.g. if there are two groups but they do not lie 180° apart from each other, they would need to be sampled). Typical examples include $-SO_3^-$, $-N(CH_3)_3^+$, and $-NO_2$.

0 do not sample these symmetric terminal groups

1 sample such terminal groups

arg2 *Ring conformation sampling*

Ring sampling is done by matching the rings in the molecules with templates for which the low-energy conformers have been previously identified. Two such template matching systems are supported:

- one using general forms for flexible 5 and 6 atom rings.
- one employing an extensive collection of specific templates for a large variety of ring systems (`ring_conf` utility).

The specific templating system has broader coverage (several hundred templates) and the template conformations were generated using MacroModel searches employing the MMFFs force field.

0 or 2 use the specific templating system (`ring_conf`)

1 use the general templating system

3 do not sample ring conformations

arg3 *Minimization of generated structures*

0 minimize generated conformations using at most the number of iterations in the `MINI` command

–1 do not minimize the generated conformations, but estimate the current energy, superimpose the generated conformations, and eliminate redundant conformations

–2 do not post process the conformations provided by ConfGen

–3 do not estimate the energies for the conformations produced but superimpose the generated conformations, and eliminate redundant conformations

>0 minimize generated conformations using at most this number of iterations

arg4 *Non-ring amide bond conformation sampling*

0 or 1 default: vary the geometry of amide bonds

2 retain the original amide bond geometry

3 make amide bond geometry trans

arg5 *Maximum relative ring conformation energy in kJ/mol*

0 use default value of 48 kJ/mol

>0 use this value (kJ/mol)

arg6 *Upper limit on the number of combinations of ring conformations sampled*

0 use default value of 16

>0 use this value

arg7 *Upper limit on the number of ring conformations sampled per ring system*

0 use default value of 8

>0 use this value

arg8 *Maximum relative ConfGen internal energy*

0 use default value of 50.0 kJ/mol

>0 use this value (kJ/mol)

CGO2 — additional ConfGen Options

CGO2 is one of a number of opcodes (CGOP, CGO2, CGO3, CGO4, CGO5, CGO6) that provide access to additional options for the ConfGen facility.

arg1 *Carboxylic acid bond conformation sampling*

0, 1 default: vary the geometry of carboxylic acid bonds

2 retain the original carboxylic acid bond geometry

3 make carboxylic acid bond geometry trans

arg3 *Method for estimating the number of degrees of freedom*

When AUOP arg5 is greater than zero, the number of conformations to sample is given by the product of this value and the number of degrees of freedom. This argument controls which method is used to estimate the number of degrees of freedom.

0, 2 Use ConfGen's internal estimate for the number of degrees of freedom, which is given by the number of rotatable bonds + $\log_2$(number of ring conformation-nitrogen atom inversion combinations)

1 Use the number of rotatable bonds given by MacroModel.

arg4 *Include the most extended conformers*

When selecting a subset of conformers from those produced by ConfGen include the arg4 most extended conformers first. In each conformer the largest distance between any two heavy atoms is calculated. The ratio of this value to the largest such distance amongst all of the conformations for this molecule is used as a measure for how extended the conformer is.

arg5 *Van derWaals radius scaling factor for close atomic approaches*

ConfGen rejects conformations with atoms closer than this factor times the sum of their van der Waals radii.

≤0 use default value of 0.6

>0 use this value

arg6 *Scaling factor for close atom gradients*

ConfGen internally minimizes structures on a simple potential function that includes a penalty term for close approaches of atoms. This argument specifies the scaling factor for this penalty term.

≤0 use default value of 1.0

>0 use this value

arg7 *Minimum van derWaals atom radius used for rejecting structures*

If the van der Waals radius for an atom is less than this value, use this value instead to determine when atoms are too close within a candidate structure.

0 use default value of 1.0 Å

>0 use this value

CGO3 — Additional ConfGen Options for enhanced sampling of weak torsional potentials

CGO3 allows the user to activate and adjust enhanced ConfGen capabilities for sampling weak torsional potentials. CGO3 is one of a number of opcodes (CGOP, CGO2, CGO3, CGO4, CGO5, CGO6) that provide access to additional options for the ConfGen facility.

sp^2–sp^3 bonds often have only two potential minima and usually have low barriers to rotation. As a result, environmental factors can force such bonds to adopt conformations that do not closely resemble either minimum. This is a concern when trying to reproduce the bioactive conformations of ligands. CGO3 activates functionality that detects rotatable bonds with low barriers to rotation and replaces the original torsional potential with a simple cosine function with a user-specified periodicity. The cosine function is shifted so that one of its minima coincides with the deepest minimum in the original torsional potential. ConfGen then systematically samples all combinations of minima in all torsional potentials including the new artificial minima for the cosine functions.

If CGO3 is not present then the original weak potential and its minima are used. The presence of CGO3 activates replacement of weak potentials by cosine functions.

arg1 *Cosine function frequency*

Number of times the cosine function repeats when rotating by 360 degrees. This also is the number of minima spaced 360/arg1 degrees apart that will be sampled.

0 use the default frequency: 6 (spacing of 60°).

> 0 use this frequency

arg5 *Energy range*

The difference in energy between the minima and the maxima in the cosine potential. This energy range is used to determine which torsional potentials are weak. If the extremes of the original torsional potential all lie within arg5 of each other then the original potential is deemed weak and is replaced by a cosine function.

0 use the default energy range 12 kJ/mol

> 0 use this value (kJ/mol)

arg6 *Maximal original energy for new minima*

All arg1 minima of the cosine potential are normally sampled by ConfGen. Some of these minima may correspond to relatively high potential regions of the original potential. The value in arg6 sets an upper bound on the energy difference between the lowest energy in the original potential and the energy in the original potential corresponding to a minimum in the cosine potential. If that bound is exceeded then this minimum in the cosine function is not sampled.

0 do not eliminate minima

>0 use this value as the upper bound on the energy difference (kJ/mol)

arg7 *restraining potential prefactor*

If the conformations are being minimized inside MacroModel itself, the dihedrals with weak torsional potentials have a periodic flat-bottomed cosine restraining potential applied. This value is the prefactor for the cosine potential.

0 use 1000 kJ/mol

> 0 use this value (kJ/mol)

arg8 *Half-width of the flat-bottom potential*

If the conformations are being minimized inside MacroModel itself, the dihedrals with weak torsional potentials have a periodic flat-bottomed cosine restraining potential applied. This value is the half width of the flat portion of the potential.

0 use 10 degrees

> 0 use this value (degrees)

CGO4 — Additional ConfGen Options for rejecting conformations with close contacts between charged functional groups

CGO4 allows the user to activate and adjust enhanced ConfGen capabilities for rejecting conformations that have formally charged functional groups that lie too close to each other. CGO4 is one of a number of opcodes (CGOP, CGO2, CGO3, CGO4, CGO5, CGO6) that provide access to additional options related to this facility.

ConfGen inherently generates conformations without considering electrostatics. This can occasionally lead to conformations that have functional groups with a formal charge of the same sign lying near each other. If minimization or energy filtering of conformations is not done inside MacroModel itself (CGEN arg3 = −1 or −2, respectively) these conformations may end up being saved in the output structure file. CGO4 turns on a penalty mechanism that can exclude such conformations within ConfGen itself. The penalty for conformation k is calculated from the fractional formal charges q_i on the atoms using the expression

$$P_k = \sum_{ij} p_{ij}$$

where the sum runs overall all pairs of atoms i and j that are separated by at least 3 bonds, and

$$\begin{aligned} p_{ij} &= 0, & q_i q_j \le 0 \\ &= A q_i q_j \exp(-r_{ij}^2/2s^2), & q_i q_j > 0 \end{aligned}$$

where A and s are constants and r_{ij} is the distance between atoms i and j.

If $P_k - P_{\min} > P_{\text{cutoff}}$ where $P_{\min}$ is the smallest penalty for any of the conformations then the conformation is eliminated within ConfGen.

arg5 *Standard deviation of the Gaussian penalty, s*

0 use 5 Ang

> 0 use this value (Ang)

arg6 *Maximum value for the Gaussian, A*

0 use 2.5

> 0 use this value

arg7 *Relative penalty cutoff, P_{cutoff}*

0 use 0.75

> 0 use this value

CGO5 — Additional ConfGen Options for rejecting conformations in which postive functional groups are close to polar hydrogens

CGO5 allows the user to activate and adjust enhanced ConfGen capabilities for rejecting conformations that have polar hydrogen atoms too close to groups with positive formal charges. CGO5 is one of a number of opcodes (CGOP, CGO2, CGO3, CGO4, CGO5, CGO6) that provide access to additional options for the ConfGen facility.

ConfGen inherently generates conformations without considering electrostatics. This can occasionally lead to conformations that have functional groups with a hydrogen bond close to and pointing towards a functional group with a positive formal charge. These conformations may end up being saved in the output structure file particularly if minimization or energy filtering of conformations is not done inside MacroModel itself (CGEN arg3 = –1 or –2, respectively) these conformations may end up being saved in the output structure file. CGO5 turns on a penalty mechanism that can exclude such conformations within ConfGen itself. The penalty for conformation k is calculated from the fractional formal charges q_i on the atoms using the expression

$$P_k = \sum_{ij} p_{ij}$$

where the sum runs over all all pairs of polar hydrogen bonds i and positively charged atoms j. Polar hydrogen bonds involve a Hydrogen atom (H) bonded to a O or N atom. The O or N atom (ON) must be at least three bonds away from atom j. The penalty term is defined by

$$p_{ij} = p_{Hj} - p_{ONj}$$

where p_{Hj} and p_{ONj} have the same functional form,

$$p_{Xj} = A q_j \exp(-r_{Xj}^2 / 2s^2)$$

A and s are constants and r_{Xj} is the distance between atoms X and j.

If $P_k - P_{\min} > P_{\text{cutoff}}$ where $P_{\min}$ is the smallest penalty for any of the conformations then the conformation is eliminated within ConfGen.

arg5	*Standard deviation of the Gaussian penalty, s*
0	use 5.0 Å
> 0	use this value (Å)

arg6 *Maximum value for the Gaussian, A*

0 use 1

> 0 use this value

arg7 *Relative penalty cutoff,* P_{cutoff}

0 use 0.25

> 0 use this value

CGO6 — Additional ConfGen Options for rejecting conformations with close contacts between nonhydrogen atoms

CGO6 allows the user to activate and adjust enhanced ConfGen capabilities for rejecting conformations that have too many nonhydrogen atoms in close proximity. CGO6 is one of a number of opcodes (CGOP, CGO2, CGO3, CGO4, CGO5, CGO6) that provide access to additional options for the ConfGen facility.

Bioactive conformations tend to be relatively extended. ConfGen can generate conformations in which topologically distant parts of the ligand lie close to each other. These conformations may end up being saved in the output structure file. CGO6 turns on a penalty mechanism that can exclude such conformations within ConfGen itself particularly if the groups approaching each other are rings. The penalty for conformation k is calculated using the expression:

$$P_k = \sum_{ij} p_{ij}$$

where the sum runs over all all pairs of nonhydrogen atoms *i* and *j* that are at least three bonds away from each other. The penalty term is defined by

$$p_{ij} = A \exp(-r_{ij}^2 / 2s^2)$$

A and *s* are constants and r_{Xj} is the distance between atoms *X* and *j*.

If $P_k - P_{\min} > P_{\text{cutoff}}$ where $P_{\min}$ is the smallest penalty for any of the conformations then the conformation is eliminated within ConfGen.

arg5 *Standard deviation of the Gaussian penalty, s*

0 use 2.5 Å

> 0 use this value (Å)

arg6 *Maximum value for the Gaussian, A*

0 use 1

> 0 use this value

arg7 *Relative penalty cutoff,* P_{cutoff}

0 use 1.00

> 0 use this value

CHYD — Suppress Hydrogen Bond Electrostatics

The presence of this opcode suppresses electrostatic interactions between the charges arising from the bond dipole between the H and donor atom (O, N or S atom) and acceptor atom (N, O, S or F) within a ligand. The exception is that sp^2 nitrogen atoms with 2 or 3 non-hydrogen attachments or nitrogen atoms that carry a positive formal charge cannot be considered acceptors.

Intramolecular hydrogen bonds in ligand-sized molecules can have a strong effect on the relative energies of conformations and yield compact conformations as the global minimum energy state. This effect is partially mitigated by using solvation treatments such as a distant dependent dielectric or GB/SA. However, intramolecular hydrogen bonds still can have an significant influence. In a protein environment, intra ligand hydrogen-bonding competes with ligand-protein hydrogen bonding. In addition, ligand conformations in ligand protein complexes are usually extended rather than compact. Thus when modeling just the ligand it can be advantageous to suppress intra-ligand hydrogen bonding interactions to better imitate ligand conformations within ligand-protein complexes. Since the electrostatic contributions dominate hydrogen bond energetics and eliminating excluded volume effects can lead to problematic behavior, the `CHYD` opcode simply turns off the electrostatic portion of the hydrogen bonding interaction. `CHYD` does not affect the electrostatics of donor and acceptor atoms if they interact via a dihedral angle potential.

arg1 *Control hydrogen-bond electrostatics*

−1 Use normal hydrogen-bond electrostatics.

0,1 Suppress hydrogen-bond electrostatics.

MINTA

5.1 Overview

One of the key issues in chemistry is chemical stability. If chemical stability could be reliably predicted by computational methods, then real molecular engineering could be achieved, and one could design stable molecules or molecular complexes having desirable properties rationally. Chemical stability is measured by the free energy of a molecule or molecular complex. The prediction of the relative free energies of different molecular systems is one of the most sought after hopes of computational chemistry. For example, in the pharmaceutical industry chemists often conceive of dozens of molecules they might synthesize but have trouble deciding which ones have the best chance of being potential drug candidates based on their own stability or the stability of molecular complexes they form. Another field of interest is chiral recognition where understanding the stability of molecular complexes involving a chiral reagent can lead to novel chiral resolution techniques. Computational techniques that help the chemists select the most promising candidates for synthesis, or design resolution techniques are extremely valuable. Unfortunately, the thermodynamics of chemical stability is quite complex, and use of a state-of-the-art arsenal of computational methods to predict the free energy of molecular systems requires very long computer simulations.

For calculations of the relative binding energies of different ligand molecules for a given receptor to work properly, many different things have to be done correctly. In particular, the gas phase potential energy force field has to be accurate, the effect of solvent has to be included in some realistic and efficient way, and all the vibrational and configurational or conformational states of the system have to be sampled with the correct Boltzmann weights. This last issue is known as the sampling problem, and is particularly difficult to solve because flexible molecules and ligand-receptor complexes may exist in many different conformations. Furthermore, these different conformations may be separated by large energy barriers that prevent them from being inter-converted using traditional simulation methods. An alternative approach embodied in the MINTA software should provide a solution to this problem by affording a direct method for the calculation of conformational and binding free energies without the need for expensive free-energy simulations.

The MINTA software incorporates new computational methodology for the direct calculation of free energy without the need for free-energy simulations and the application of "computational alchemy." MINTA utilizes a basic assumption that the total free energy of small to medium sized molecules and molecular complexes is comprised mainly of contributions from

the low-energy wells of their respective potential energy surfaces (PES). MINTA relies upon an exhaustive conformational search of the low-energy minima, and its basic tenet is a novel multidimensional integration technique that allows, for the first time, for the numerical integration of the high-dimensional configuration integral of individual energy wells. MINTA can, perhaps, be best described in contrast to the well-known quasi-harmonic approximation. Both methods recognize that the local thermodynamics of an energy well can be described by a Boltzmann distribution function. The essence of the quasi-harmonic approximation is to estimate an effective Hessian **H** by calculating the co-variance matrix of the internal coordinate variations during a short molecular dynamics (MD) simulation that is local to a particular energy well. The resulting **H** is not equal to the true Hessian **H**, because the effective Hessian includes, implicitly, some anharmonic effects due to the MD simulation. Nevertheless, **H** is used in the context of the harmonic oscillator model to estimate entropy and conformational free energy. MINTA operates exactly the other way around. Instead of sampling the real PES in order to generate an effective Hessian, MINTA utilizes the real Hessian to sample the PES efficiently. The resulting MINTA integrals of the individual energy wells are, then, summed together to calculate the total molecular configuration integral and the total free energy.

Currently, MINTA can be used for fast computation of the conformational free energy of small and medium sized molecular systems *in vacuo* and in the presence of a continuum solvent model. In particular, MINTA is an excellent tool for the calculation of the binding free energy of molecular complexes comprised of substrate molecules bound to small receptors used in the molecular recognition field or enzyme receptor models often used in pharmaceutical research. Unlike available free-energy simulation programs, MINTA calculations are extremely user-friendly and should find wide utility as a simple tool for medicinal chemists already familiar with conformational analysis.

5.2 Introduction to MINTA

The MINTA methodology is introduced here in the context of calculating binding free energies. The statistical-thermodynamic foundation of the calculation of binding affinities of molecular complexes is quite complex, but for most practical problems, the stability of host-guest complexes can be formulated in terms of binding free-energy (BFE) differences. There can be various levels envisioned at which approximations to BFE differences can be made. For example, one wishes to calculate the BFE difference ($\Delta\Delta G_{L\text{-}D} = \Delta G_L - \Delta G_D$) between the L and D enantiomers of a ligand bound to an enantioselective host. The simplest approach one can follow is to calculate the energy difference between the lowest energy L and the lowest energy D enantiomer of the ligand. Of course, this approach ignores entropic effects due to the fact that first of all there are multiple binding conformations of both the L and D enantiomers and second of all, the individual conformations are not static (confined to the bottom of their energy well) but exhibit large dynamic diversity in terms of conformational changes limited to

that energy well. Note that there can be numerous low-energy binding conformations of both enantiomeric states.

The next level of approximation to the BFE is the inclusion of multiple conformations. With this model, the multiple low-energy binding conformations of the two enantiomers of the ligand are considered as two sets of discrete energy levels corresponding to the potential energy of the individual binding conformations. A simple statistical mechanics calculation can then be used to estimate the binding free-energy difference between the L ligand and the D ligand with respect to the enantioselective host. One can also include some of the vibrational and the rotational free energy utilizing the well-known rigid-rotor harmonic quantum oscillator (RRHO) model. The ultimate approach, in the classical sense, for calculating BFE differences, however, involves the computation of the molecular partition function often termed molecular configuration integral.

For enantioselective binding, for example, the direct calculation of $\Delta\Delta G_{\text{L-D}}$, again in the classical sense, involves the evaluation of the molecular configuration integral Q:

$$Q_{\text{L}} = \sum_{i=1}^{n_{\text{L}}} \int_{V_i^{\text{L}}} e^{-\frac{E(\mathbf{r})-E_0}{RT}} d\mathbf{r}, \qquad Q_{\text{D}} = \sum_{i=1}^{n_{\text{D}}} \int_{V_i^{\text{D}}} e^{-\frac{E(\mathbf{r})-E_0}{RT}} d\mathbf{r} \tag{1}$$

$$\Delta G_{\text{L-D}} = -RT \ln \frac{Q_{\text{L}}}{Q_{\text{D}}} \tag{2}$$

It is assumed with the use of the MINTA software that the dominant part of the configuration integral comes from contributions at or near to low-energy binding conformations. Conformational search results on ligand-receptor complexes suggest that this approximation is feasible for binding free-energy calculations. Therefore, Q is summed over, respectively, n_{L} and n_{D} conformations each encompassing different V volumes of the conformational space. Note that the individual terms of the two sums in equation 1 include all the vibrational and configurational states of the particular conformational energy wells of L and D conformations, respectively. Also note that all of the symmetry related copies of a single L or D conformation should be included in the sum in equation 1 to account for the statistical correction for conformational symmetry. $E(\mathbf{r})$ is the molecular mechanics energy with respect to the nuclear coordinates $\mathbf{r}$. $E(\mathbf{r})$ includes the solvation energy as well, preferably in terms of a continuum model which does not introduce new degrees of freedom by explicit solvent molecules. E_0 is the global minimum energy, which is the common reference for both L and D binding conformations. Thus, E_0 could refer to the lowest energy L or the lowest energy D conformation, whichever is lower. R is the gas constant and T is the absolute temperature.

Direct evaluation of the configuration integral has been considered to be impossible to solve except for problems of very low dimensionality. Instead, indirect methods utilizing various simulation techniques based on free-energy perturbation (FEP) have found widespread utility. These methods belong to the realm of molecular dynamics and Monte Carlo simulations using explicit solvent models. In Appendix G a brief introduction is provided to this area to put MINTA in perspective in the world of free-energy simulations.

The basic tenet of the MINTA methodology is a novel Monte Carlo integration technique termed mode integration (this is where the name MINTA comes from). The MINTA software allows, for the first time, the direct calculation of the configuration integral of single molecules and molecular complexes of real chemical interest, without the need for expensive free-energy simulations and the use of "computational alchemy."

In the current context, ΔG represents the MINTA Free Energy, which only includes the enthalpy (MM energy) and configurational entropy due to different conformations and the shape of individual conformational energy wells in a normal mode coordinate system, ignoring translation and rotation. Typically, in our examples, the estimated ΔG's correspond to the standard state free-energy, ΔG_0, within the MINTA approximations.

5.3 MINTA Methodology

The patented MINTA methodology addresses the sampling problem for calculating free energies at two different levels. First, a global conformational search is carried out to identify the low-energy regions of the potential energy surface (PES), which correspond to the low-energy conformations of a single molecule or the low-energy binding conformations of a molecular complex. Local sampling of each individual conformational energy well is then accomplished utilizing the mode integration technique embodied in the MINTA software. MINTA can, perhaps, be best described in contrast to the well-known quasi-harmonic approximation. Both methods recognize that the local thermodynamics of each energy well i can be described by a Boltzmann distribution function, which is—in the harmonic approximation—a normalized multivariate Gaussian distribution function:

$$p_i = \sqrt{\frac{\det \mathbf{H}_i}{(2\pi RT)^n}} \exp\left(-\frac{1}{2RT}(\mathbf{r}-\mathbf{r}_i)\mathbf{H}(\mathbf{r}-\mathbf{r}_i)\right) \tag{3}$$

where $\mathbf{r}_i$ and $\mathbf{H}_i$ denote, respectively, the bottom of a particular energy well and the associated local Hessian matrix. The Hessian is evaluated at the bottom of the well. The number of degrees of freedom n is equal to the number of unconstrained internal coordinates, which include the six relative translation-rotational degrees of freedom defining the relative orientation of the host and the guest in a molecular complex. R is the gas constant and T is the temper-

ature. Note that $\mathbf{H}_i$, from the statistical point of view, is none other than the inverse co-variance matrix of the corresponding Gaussian distribution.

The essence of the quasi-harmonic approximation is to estimate an effective Hessian $\mathbf{H}_i$ by calculating the co-variance matrix of the internal coordinate variations during a short molecular dynamics (MD) simulation that is local to energy well *i*. The resulting $\mathbf{H}_i$ is not equal to $\mathbf{H}_i$. The effective Hessian includes, implicitly, some anharmonic effects due to the MD simulation. Nevertheless, $\mathbf{H}_i$ is used in the context of the harmonic oscillator model to estimate entropy and conformational free energy. MINTA operates exactly the other way around. Instead of sampling the real PES in order to generate an effective Hessian, MINTA utilizes the real Hessian to sample the PES efficiently. However, MINTA sampling is carried out in the context of multidimensional Monte Carlo integration, not some free-energy simulation, to afford a unique, direct method for calculating conformational and binding free energies. Mode integration means, in essence, that equation 3 is utilized as a sampling function to integrate equation 1 in *normal mode space*. For further details, see [42, 43, 44, 45].

5.4 MINTA Commands

The MINTA software is extremely user-friendly. MINTA is launched from Maestro via a seamless interface that allows the user to invoke MINTA by a single command from a familiar MacroModel command file. The MINTA command has been structured to conform with current MacroModel syntax. A basic sample MINTA command file is shown here:

```
minta.mae
minta-out.mae
 FFLD
 BGIN
 READ
 MINI
 MNTA
 END
```

For users familiar with MacroModel this command structure is well-known, running a so-called multiple minimization job. The input file `minta.mae` should contain one or more conformations of the *same* molecule or molecular complex. The `BGIN`/`END` loop reads the structures one by one from the input file and minimizes the energy of the structures with the force field selected in the `FFLD` command. The `MNTA` command carries out the MINTA calculation on every minimized structure. The output file `minta-out.mae` will contain the minimized structures in the order they were read in.

MNTA — Mode iNTegrAtion

arg1 *Number of MINTA iterations*

The MINTA numerical integrals are calculated in statistical blocks to achieve better convergence. The number of blocks used is referred to as the number of MINTA iterations.

5 (Default).

n $n > 0$ (It is *not* recommended to use values $n > 10$).

arg2 *Number of energy evaluations per MINTA iteration*

MINTA integration is based on *single point* energy evaluations. The total number of energy evaluations *per structure* is equal to $arg1 \times arg2$.

0 2000 (default).

n $n > 0$ (It is *not* recommended to use values $n > 10{,}000$).

arg3 *Flag to choose adaptive MINTA integration*

Adaptive MINTA integration is slightly more accurate than the default, non-adaptive integration mode. Note however, that using adaptive MINTA is expected to be beneficial *only* if *arg4* is in the range of 1-10.

0 Non-adaptive MINTA (default).

n $n \neq 0$ selects adaptive MINTA (*only* recommended when $1 \leq |arg4| \leq 10$).

arg4 *Number of "soft" degrees of freedom, for which numerical MINTA integration is applied*

MINTA integration is carried out in normal mode space. Although MINTA can be instructed to utilize numerical integration in all degrees of freedom, it is far more efficient to partition the degrees of freedom into two categories, "soft" and "hard," corresponding to low-frequency and high-frequency vibrational modes, respectively. The partition is somewhat arbitrary, of course, but generally "hard" modes can be integrated with high accuracy using an extremely fast analytical approximation to the MINTA integral. "Soft" modes, however, have to be integrated numerically to account for significant anharmonic effects. Therefore, a typical MINTA calculation involves numerical integration in "soft-mode" space and analytical integration in "hard-mode" space.

0 Numerical integration in *all* degrees of freedom (default). Only recommended for small molecules of up to twenty atoms.

n For $|n| > 0$, $|n|$ "soft" modes will be used and the rest of the degrees of freedom will be treated as "hard." For $n > 0$, the "hard" modes will be integrated utilizing the fast, analytical MINTA approximation. For $n < 0$, the "hard" modes will be integrated utilizing the traditional harmonic oscillator model.

Note that arg3 should always be zero unless $1 \leq |n| \leq 10$. Also note that the user is strongly cautioned *not* to use values $|n| > 50$.

arg5 *Temperature (K)*

0 300 (default).

x $x > 0$.

arg6 *Hard limit for sampling along normal modes (Å)*

Sampling will be limited to this distance from the equilibrium geometry of the structure along any of the normal mode directions in $3N$-6(5) dimensional normal mode space where N is the number of atoms.

0 1 Å (default).

x $x > 0$ (it is *not* recommended to use values $x > 3$).

arg7 *Soft limit for sampling along normal modes (units of standard deviation)*

Sampling will be limited to different distances from the equilibrium geometry along different normal mode directions. For a particular mode i sampling is limited to a particular distance, which is equal to *arg7*-times the standard deviation of the multi-dimensional Gaussian function in equation 3, along the particular normal mode direction i. The value of distance i in Å is $arg7 \times \sqrt{(RT/\lambda_i)}$ where λ_i is the ith eigenvalue of the Hessian matrix. Note that the softer the mode the larger the sampling distance because of the reciprocal nature of λ_i in the standard deviation formula.

Important: arg6 takes precedence over arg7. The actual sampling distance along a particular normal mode i will be MIN(arg6, $\text{arg7} \times \sqrt{(\text{RT}/\lambda_i)}$). This is an important safeguard to prevent incorrect sampling due to artificially small eigenvalues associated with extremely soft, highly anharmonic vibrational modes.

0 3 units of standard deviation (default).

x $x > 0$ (it is *not* recommended to use values $x > 5$).

arg8 *Not in use*

DEBG switch 1001 prints details of the numerical integration in the logfile. The calculated numerical integrals are listed for each statistical block (MINTA iteration) individually and the block averages are also printed along with their χ^2 test values. Any significant discrepancy of χ^2 from 1 indicates insufficient sampling and consequently, the MINTA results cannot be trusted. Insufficient sampling can also be detected by looking at the error bar associated with each integral value. The error bar is defined as ± 1 standard deviation. The debugging output also includes the actual size of the integration box in "soft-mode" space.

5.5 Notes

1. The input file of a MINTA calculation should contain the output of a preceding MacroModel conformational search.
2. The conformational search should not discard symmetrically equivalent conformations via NSRO, NSRF, ATEQ, NSEQ, and MSYM commands in order for their statistical weights to be accounted for correctly.
3. The energetic parameters, energetic constraints, and substructure definitions in the MINTA command file (FFLD, SOLV, EXNB, CHGF, FXDI, FXBA, FXTA, SUBS, FXAT) should be identical to those used in the preceding conformational search.
4. MINTA free energy should be looked at in exactly the same way as the molecular mechanics energy.
   - The MINTA free energy is only meaningful when comparing the difference between two conformations of the same molecule. It estimates the free-energy difference between those two conformations.
   - In the same way, MINTA can be used to calculate the free-energy difference between two molecules whose molecular mechanics energy is comparable (typically stereo-isomers of any kind).
   - For molecules with non-comparable molecular mechanics energy, MINTA has to be applied in terms of a thermodynamic cycle in order to estimate, for example, the binding free-energy difference between two different ligands bound to the same receptor (see Appendix G).

Note: The MINTA software calculates the MINTA free energy of each conformation in a multi-conformer input file, but MINTA also calculates the *total free energy* of the whole input file, i.e., the total free energy of the whole set of structures in the input file. *The total free energy of different multi-conformer files can be compared using the exact same criteria applied to individual conformations.* For example, the results of a

conformational search on glucose are separated in two multi-conformer files, α anomers in one file and β anomers in a second file. Two MINTA calculations carried out on the two different files will result in the total free energy of the α anomers and the β anomers, respectively. The difference between the α and β total free energies provides an estimate for the measurable, so-called anomeric free energy of glucose.

5. As a rule of thumb, the CPU time required to run a nearly complete conformational search on any kind of molecular system is comparable to the CPU time required running a subsequent MINTA calculation.
6. *Finally, it should be borne in mind that MINTA is only as good as the force field and the conformational search.* MINTA is expected to provide a good estimate of the free energy of a molecular system for a given force field, but MINTA is always subject to serious error due to inadequate force field selection or an incomplete conformational search.

5.6 Working Examples

Some examples of MINTA applications, including the `.log` files, can be found in the directory `$SCHRODINGER/macromodel-v`*version*`/test/minta`. These examples are discussed below.

5.6.1 Cyclononane

A very instructional example is the MINTA calculation on cyclononane. In this example we look at the free energies of the individual conformers of cyclononane. Cyclononane has seven low-energy conformers within 50 kJ/mol above the global minimum on the MM2 potential energy surface. The global minimum is a highly symmetrical (C_3), deep minimum, which is 3.14 kJ/mol deeper than the second lowest energy minimum. However, it has been suggested that entropic effects due to the high flexibility of the second and third lowest energy minima could account for decreasing the *free-energy gap* between the global minimum and higher minima. In fact, a simple MINTA calculation suggests that not only a decrease in the free-energy gap is predicted, but the (free) energetic order is turned around. *Such (free) energetic reordering of conformations with respect to their energetic order based on steric energy only is an extremely important aspect of molecular design where stable conformations of a molecule are sought.*

In symmetrical molecules, like cyclononane, the statistical weight (due to the existence of symmetry related copies of each conformation) has a significant contribution to the free energy of a particular conformation. There are at least two ways to deal with this. One way is to apply a statistical correction to the free energy that one might calculate for a single copy of each individual conformation. However, with MINTA, we generally recommend using the following (somewhat redundant) alternative:

First, a conformational search is performed, in which neither the NSRO or MSYM commands are used (see Note 2 on page 172) *and hence, the symmetry related copies are not discarded*. Then, the single output file resulting from the conformational search is separated into seven individual files, each containing multiple copies of the same conformation.

Thus, in the subdirectory c9 the following files are provided for running the MINTA calculation on cyclononane:

- *n*.mae where *n* stands for the numbers 1-7. These multi-conformer MacroModel input files contain multiple copies of the seven conformations of cyclononane found by a preceding conformational search. As stated above, the conformational search was run with MacroModel *without* the NSRO or MSYM commands to keep all of the symmetrical copies of the same conformation, subject only to a numbering system rotation around the nine-membered ring. For an asymmetrical conformation this results in nine copies (conformations 2, 3, 6, and 7). The global minimum has C_3 symmetry, therefore, it has only three symmetrical copies due to the numbering system rotation. Conformations 4 and 5, on the other hand, do not have nine but eighteen copies. MacroModel normally excludes enantiomers from a conformational search, however, conformations 4 and 5 of cyclononane are special in that they are not only equivalent but *identical* to their enantiomers. This is why they both have eighteen copies, not nine. In summary, the statistical weight of the global minimum is only one third of conformations 2, 3, 6, and 7 and one ninth of conformations 4 and 5.

- c9_minta.com, the MINTA command file:

```
c9_minta.dat
c9_minta.out
 DEBG    1001
 DEBG     601
 EXNB       0      0      0      0     0.0000     0.0000     0.0000     0.0000
 FFLD       1      0      0      0     0.0000     0.0000     0.0000     0.0000
 BGIN
 READ
 MINI       9      0     50      0     0.0000     0.0000     0.0000     0.0000
 CONV       2      2      0      0     0.0000
 MNTA       5   5000      0     50   300.0000     1.0000     3.0000     0.0000
 END
```

  Note that the individual dat-files have to be renamed c9_minta.dat to be able to run this particular MINTA command file. DEBG 1001 will print the MINTA debugging output, DEBG 601 prevents MINTA from writing out the minimized structures after minimization. The structures will be written to the output file after the MINTA calculation is completed and their MINTA free energy will be printed in the title line of the structure in c9_minta.out.

- The MINTA calculation runs five iterations with 5,000 energy evaluations in each itera-

tion (25,000 total per conformation). Non-adaptive Monte Carlo integration is applied in "soft-mode" space spanned by the first 50 low-frequency vibrational modes. The rest of the modes are integrated using the fast analytical MINTA method. The temperature is set to 300 K. The size of the integration volume (box) in "soft-mode" space is determined by the lesser of 1 Å and three-times the standard deviation (see *arg6* and *arg7* of the MINTA command description above).

- *n*`.log` where *n* stands for the numbers 1-7. These logfiles are the logfiles of the MINTA calculations 1-7.

  Based on the 'Gtotal' values in the logfiles, the MINTA calculation on cyclononane yields the following result:

```
Order of cyclononane conformations based on (MM2) steric energy:
_
_                E      ΔE (kJ/mol)
                ------------------
        1.      97.86   0.00      global minimum steric energy
        2.     101.00   3.14
        3.     101.10   3.24
        4.     107.16   9.30
        5.     111.10  13.24
        6.     121.59  23.73
        7.     141.13  43.27

Order of cyclononane conformations based on (MM2) MINTA free energy:

                 G     ΔG (kJ/mol)
                -----------------
        1.(2.) 481.24   0.70
        2.(3.) 481.36   0.82
        3.(1.) 480.54   0.00      global minimum free energy
        4.(4.) 484.81   4.27
        5.(5.) 489.32   8.78
        6.(6.) 499.58  19.04
        7.(7.) 521.48  40.94
```

5.6.2 Glucose

In the subdirectory `glucose` the necessary files are provided for running an anomeric free-energy calculation on glucose using the AMBER* force field and GB/SA continuum solvation for water. In this example, we are looking at the total free energy of each of two sets of conformations (one set for the α anomer, and one for the β anomer). Note that the α anomer and β

anomer of glucose are diastereomers. Also note that the MINTA calculation on glucose predicts the β anomer to be more stable than the α anomer by 1.38 kJ/mol, which is within 0.04 kJ/mol of the experimental value (1.42 kJ/mol). However, the calculation is an "overkill," including far too many conformations. The user is encouraged to test how many of the approximately 1500 conformations can be omitted to maintain a reasonable accuracy in the calculation.

5.6.3 Vancomycin

In the subdirectory `complex` the necessary files are provided for running a binding free-energy calculation involving the vancomycin Nac-D-Ala-D-Ala complex using the AMBER* force field and GB/SA continuum solvation for water.

The `complex` sample calculation involves fifteen low-energy binding conformations of the vancomycin Nac-D-Ala-D-Ala and Nac-L-Ala-L-Ala complex, respectively. This example has significant relevance to pharmaceutical applications.

Antibiotics of the vancomycin group have gained increasing clinical importance during the last twenty years for the treatment of Gram-positive bacterial infections, particularly those which exhibit multiple drug resistance, as described in the literature [46, 47, 48]. The molecular basis for the mode of action of vancomycin involves the binding of vancomycin to cell-wall mucopeptide precursors terminating in the peptide -D-Ala-D-Ala. The peptide cell-wall analogue NAc-D-Ala-D-Ala has served as a convenient model for X-ray and NMR studies aimed at determining the binding mode of vancomycin. The vancomycin Nac-D-Ala-D-Ala complex represents an interesting "reverse" binding structure. The real binding structure involves the vancomycin substrate binding to the cell-wall whereas the reverse model system involves the cell-wall analogue Nac-D-Ala-D-Ala substrate binding to the vancomycin binding site. As reported by Dunayevskiy et al. [49], Vancomycin has recently been used as the model receptor in several experimental studies for the measurement of binding affinities of peptides to vancomycin. It has been experimentally determined that while the Nac-D-Ala-D-Ala ligand binds to vancomycin very tightly, the Nac-L-Ala-L-Ala ligand is only a very week binder. A qualitative/semi-quantitative MINTA calculation is presented in the `complex` sample where the difference in binding affinity is predicted to be 3.1 kcal/mol in favor of the Nac-D-Ala-D-Ala ligand. Note, however, that a fully quantitative MINTA analysis would require a significantly more complete conformational search and proof of the aptness of the AMBER* force field and GB/SA solvation. Again, the user is encouraged to further experiment with vancomycin.

MacroModel Test Suites

The MacroModel test subdirectories are located in the `$SCHRODINGER/macromodel-v`*version*`/test` subdirectory. The suite of tests in any directory can be run at your site by editing and executing the file called `do` in that directory.

You can run the standard test suite, which is in the directory `bmin.test`, using the `-TEST` argument to the `bmin` script in the `$SCHRODINGER` directory:

```
$SCHRODINGER/bmin -TEST
```

The output from this command is written to the current working directory. This is our standard test suite for program correctness. It tests correctness of energetic procedures (such as minimization) as well as of simple energy evaluations. However, it is not exhaustive: things can go wrong which are not revealed by the results of this suite. It is normal for small numerical variations to occur between versions of MacroModel compiled on different hosts, and even on the same host at different compilation levels. Larger variations can and do build up when long or random trajectories are followed, as in molecular-dynamics and conformational-search procedures. Thus, interpreting the results of these tests requires a certain amount of experience.

A program called `compare` is included in this directory. This runs the `diff` (or, on SGI, the `gdiff`) program on two sets of test-suite results, after filtering out certain irrelevant lines. The command syntax is:

`compare` *suite1 suite2*

where *suite1* and *suite2* are files containing results of the test suite to be compared.

MacroModel Benchmark Suites

A benchmark suite is now included in the MacroModel distribution for use in comparing the relative speed of different releases or of the same release on different platforms. The benchmark suite contains a number of individual tests which collectively exercise MacroModel calculations involving minimizations and conformational searches on small and protein-sized molecules.

To run the benchmark copy the directory:

`$SCHRODINGER/macromodel-v`*version*`/test/benchmarks`

to a suitable location. In the new benchmarks directory type:

`./do >` *run_log* **`&`**

At the end of the calculation a file called `Summary.`*machine_name* is written that contains the times for the various tests performed. There is also a README file in the benchmark directory that provides a brief description of the various benchmarks.

Note: Schrödinger, LLC will not provide benchmark results. If you write a request for such results to the independently administered MacroModel mailing list at `mmodlist@chem.iupui.edu` another MacroModel customer might be able to help you.

Atom and Bond Types

C.1 The atom.typ File

MacroModel atom types are not "hard-wired" into the program, but are read in at run-time from a file called `atom.typ`. We continue to use the original MacroModel atom types for most applications, but now have a mechanism for easily adding new types. The list of explicitly supported atom types has been expanded to include:

Li^{+}, Na^{+}, K^{+}, Rb^{+}, Cs^{+}, Ca^{2+}, Ba^{2+}, Mg^{2+}, Mn^{2+}, Mn^{3+}, Mn^{4+}, Mn^{5+}, Mn^{6+}, Mn^{7+}, Fe^{2+}, Fe^{3+}, Co^{2+}, Co^{3+}, Ni^{2+}, Ni^{3+}, Cu^{+}, Cu^{2+}, Zn^{2+}, Mo^{3+}, Mo^{4+}, Mo^{5+}, Mo^{6+}, and Cl^{-}.

You can add your own atom types and supply parameters for these types in the force-field file. The default `atom.typ` file is the one located in `$SCHRODINGER/mmshare-v`*version*`/data`; a local file of this name overrides the default, as does a local file whose name is *filename*`.typ`, where *filename* is the stem of the *filename*`.mae` file. This system of default version with local overrides is analogous to that used for force field and solvation files.

Currently MacroModel is parameterized to accept atom types up through number 300. We restrict the type numbers that we supply to the range 0 through 199. When adding your own types, you should begin with the number 200. Negative atom types are not possible.

The Maestro GUI also reads the `atom.typ` file, and you may change atoms in built structures to new types (including user-defined types) from the graphical interface. However, no provision is made in the current version for the builder to obtain the geometric information it needs from the `atom.typ` file, which would require extensions to the format of this file. Also, the original atom types remain "hard-wired" in MacroModel; thus, we (again) advise users to add new types if desired, but not to redefine types that we supply.

The `atom.typ` file itself contains comments describing its format and usage.

C.2 MacroModel Bond Types

The following table lists the values that may appear in the MacroModel structure file format for bond orders, and the corresponding meaning of these values.

Table C.1. Bond order values and symbols.

Value	Symbol	Description
0	.	Zero order bond
1	-	Single bond
2	=	Double bond
3	%	Triple bond
4	*	Bond of undefined order (excluding zero order)

C.3 MacroModel Atom Types

The following table lists the atom types that are used in MacroModel and their equivalencies to atom types for other force fields. These atom types are defined in the `atom.typ` file. See Section C.1 on page 181 for a description of the `atom.typ` file and how to add atom types.

Note that in MacroModel, 00 atoms are stored as type 0 in the atom connection table, and that all current force fields require explicit hydrogens on heteroatoms.

Table C.2. Atom types and equivalents.

			Equivalencies		
No.	Symbol	Description	MM2/ MM3	Charmm	Amber
1	C1	Carbon - sp			
2	C2	Carbon - sp^2	2,3	CT,C	C, C*, CA, CB, CC, CM, CF, CG, CN
3	C3	Carbon - sp^3	1	CT	CT
4	CA	United atom CH - sp^3		CH1E	CH
5	CB	United atom CH_2 - sp^3		CH2E	C2
6	CC	United atom CH_3 - sp^3		CH3E	C3
7	CD	United atom CH - sp^2		CR1E	CD, CE, CJ, CP
8	CE	United atom CH_2 - sp^2			
9	CF	United atom CH - sp			
10	CM	Carbanion (C^-)			

Table C.2. Atom types and equivalents. (Continued)

			Equivalencies		
No.	**Symbol**	**Description**	**MM2/ MM3**	**Charmm**	**Amber**
11	CP	Carbocation (C^+)	30		
12	CR	Carbon free radical	29		
...					
14	C0	Any carbon			
15	O2	Oxygen - double bond	7	O	O, O2
16	O3	Oxygen - single bond	6	OH1, OH2	OS, OH
17	OA	United atom OH		OH1E	
18	OM	O^- (alkoxide, carboxylate)	47		OC
19	OW	United atom H_2O		OH2E	
20	OP	Oxonium (sp^2) =[O^+]–			
21	OQ	Oxonium (sp^3) R_3O^+			
...					
23	O0	Any oxygen			
24	N1	Nitrogen - sp	10		
25	N2	Nitrogen - sp^2	9	N, NR, NP, NH1, NH2	N, NB, NC, N*, N2
26	N3	Nitrogen - sp^3	8	NH3	NT
27	NA	United atom NH - sp^3			
28	NB	United atom NH_2 - sp^3		NH1E	
29	NC	United atom NH - sp^2			
30	ND	United atom NH_2 - sp^2		NH2E	
31	N4	N^+ - sp^2		NC2	NA
32	N5	N^+ - sp^3	39		N3
33	NE	United atom NH^+ - sp^3			
34	NF	United atom NH_2^+ - sp^3			
35	NG	United atom NH_3^+ - sp^3		NH3E	

Table C.2. Atom types and equivalents. (Continued)

			Equivalencies		
No.	**Symbol**	**Description**	**MM2/ MM3**	**Charmm**	**Amber**
36	NH	United atom NH^+ - sp^2			
37	NI	United atom NH_2^+ - sp^2		NC2E	
38	NM	N^- - sp^3			
39	NP	N^- - sp^2			
40	N0	Any nitrogen			
41	H1	H-Electroneut(e.g., C,S)	5, 44	HA	HC, HS
42	H2	H-O (Neutral)	21, 24	H,HC	HO
43	H3	H-N (Neutral)	23, 28	H,HC	H, H2
44	H4	H^+ (Cation)	48 MM3 (none in MM2)		H3
45	H5	H^- (Anion)			
...					
48	H0	Any hydrogen			
49	S1	Sulfur	15,17,18	S	S, SH
50	SA	United atom SH		SH1E	
51	SM	S^- (thiolate anion)			
52	S0	Any Sulfur		S	
53	P3	Phosphorus, trivalent	25		P
54	B2	Boron - sp^2	26		
55	B3	Boron - sp^3	27		
56	F0	Fluorine	11		
57	Cl	Chlorine	12		
58	Br	Bromine	13		
59	I0	Iodine	14		
60	Si	Silicon	19		
61	Du	Dummy atom for FEP			

Table C.2. Atom types and equivalents. (Continued)

			Equivalencies		
No.	**Symbol**	**Description**	**MM2/ MM3**	**Charmm**	**Amber**
62	Z0	Special atom to be defined			
63	Lp	Lone electron pair			LP
64	00	Any atom	0	*	
65	Li	Li^{+}			Li
66	Na	Na^{+}		SOD	Na
67	K0	K^{+}		POT	K0
68	Rb	Rb^{+}			Rb
69	Cs	Cs^{+}		CES	
70	Ca	Ca^{+2}		CAL	
71	Ba	Ba^{+2}			
72	Mg	Mg^{+2}		MG	
73	M2	Mn^{+2}			
74	M3	Mn^{+3}			
75	M4	Mn^{+4}			
76	M5	Mn^{+5}			
77	M6	Mn^{+6}			
78	M7	Mn^{+7}			
79	f2	Fe^{+2}			
80	f3	Fe^{+3}			
81	o2	Co^{+2}			
82	o3	Co^{+3}			
83	n2	Ni^{+2}			
84	n3	Ni^{+3}			
85	c1	Cu^{+}			
86	c2	Cu^{+2}			
87	Zn	Zn^{+2}			
88	m3	Mo^{+3}			

Table C.2. Atom types and equivalents. (Continued)

			Equivalencies		
No.	**Symbol**	**Description**	**MM2/ MM3**	**Charmm**	**Amber**
89	m4	Mo^{+4}			
90	m5	Mo^{+5}			
91	m6	Mo^{+6}			
92	Sr	Sr^{+2}			
93	L0	Lithium neutral			
94	M0	Magnesium neutral			
...					
100	SP	S^{+}			
101	S2	Sulfur - sp^2			
102	Cm	Cl^{-}		CLA	
103	B0	Any boron			
104	Fm	F^{-}			
105	Bm	Br^{-}			
106	Im	I^{-}			
107	P5	Phosphorus, pentavalent tetrahedral (e.g. PO_4^{3-})			
108	P0	Any phosphorus			
109	S4	Sulfur, tetravalent			
110	S6	Sulfur, hexavalent octahedral (e.g. SF_6)			
111	P4	P^{+}, tetravalent			
112	Se	Selenium			
113	ST	Sulfur, hexavalent tetrahedral (e.g. SO_4^{2-})			
114	Sm	Sulfide anion, S^{2-}			
115	Om	Oxide anion, O^{2-}			
...					
150	PI	Ligand dummy atom			

Table C.3. Generalized atom types.

No.	Symbol	Description
151	GA	Isolated atom
152	GB	Linear-single coordinate
153	GC	Linear-two coordinate
154	GD	Trigonal-two coordinate
155	GE	Trigonal-three coordinate
156	GF	Tetrahedral-three coordinate
157	GG	Tetrahedral-four coordinate
158	GH	Trigonal bipyramid-three coordinate
159	GI	Trigonal bipyramid-four coordinate
160	GJ	Trigonal bipyramid-five coordinate
161	GK	Octahedral-four coordinate
162	GL	Octahedral-five coordinate
163	GM	Octahedral-six coordinate
164	GN	Pentagonal bipyramid-seven coordinate
165	GO	Twisted cube-eight coordinate
166	GP	Nine coordinate
167	GQ	Ten coordinate
168	GR	Eleven coordinate
169	GS	Icosahedron-twelve coordinate
170	GT	Thirteen coordinate
171	GU	Fourteen coordinate
172	GV	Fifteen coordinate
173	GW	Sixteen coordinate

Force-Field File Format

D.1 Organization of the Force-Field File

Force-field parameters are obtained by MacroModel at execute time prior to performing energy calculations. Force-field parameters may be obtained from a co-process, using the BMFF mechanism (see Appendix E) or, more commonly, may be read in from a file, the force-field file. This appendix describes the latter mechanism. The force-field file tells the program which of the various possible energetic equations the particular force field uses for each type of interaction, and what the values of the parameters are (force constants, natural values of internal coordinates, etc.) for different combinations of atoms. Making changes to the parameter set used by a calculation is simply a matter of editing the force field file to change parameters or adding new ones.

See the description of the `FFLD` command for more information about the force fields we supply.

Force-field files have several sections:

- The *Introductory Section* tells the program which equations and conversion factors to use in computing energies. This section also lists a variety of alternatives and selections which may be chosen by making minor editing changes to the file. Finally, special atom symbols may be defined that will be equivalenced to multiple standard atom types for the convenience of further specification.
- The *Main Parameter Section* contains parameters which can be assigned based on knowing the local environment of the atoms involved in an interaction (a stretch, bend, torsion or nonbonded atom pair) out to at most a distance of two covalent bonds.
- The *Substructure Section* contains parameters which can be assigned only by knowing how atoms in an interaction are placed in a larger context; for example, some AMBER* parameters for atoms in standard amino acids are assigned differently in this section depending on which amino acid the atom is part of.

At the left of each line is a two-character field which indicates the format or contents of the line. This two-character field is most often interpreted as an integer, `I`, though in certain situations it contains character data. If `I` is a negative number, it indicates the Fortran format for reading the group of subsequent lines, as described in Table D.1.

Table D.1. Force field file line formats.

I	Format	Description
-1	I2,4X,A3,4X,I2,4X,F11.5	Energy equation switches
-2	I2,2X,3(A2,1X,A1,1X),A2,3X,3(F9.4,2X),4(2(A2),1X)	Normal interactions
-2	I2,2X,4(A2,2X),4X,3(F9.4,2X)	Special substructure interactions
-3	I2,2X,75A1	Special substructure linear notation format
-4	I2,10F9.4,5(1X,A1),1X,I3,1X,A1,A30	Special substructure atomic charges
-5	I2,2X,A2,2X,29(A2,1X)	Atom equivalencing specifications
-6	2X,A2,2X,6(F9.4,2X),2A2,1X,A2	VDW interactions and single atom charges

`I=-2` has different meanings depending on whether the lines in question occur in the Main Parameter Section or the Substructure Section of the force field. To the end of these formats is appended

`,2X,5(1X,A1),1X,I3,1X,A1,A30.`

This addition allows reading of alternate and selection information relating to the quality and origin of the parameters.

Values of `I` greater than or equal to zero or containing character data specify the nature, rather than the format, of the data to be read, the format having already been specified by a preceding line with `I<0`. The meaning of these values of `I` is discussed in the appropriate section below.

D.2 Introductory Section

BMFF force-fields have several lines at the very top giving MacroModel information about the coprocess to be launched. These are discussed in Appendix E.

An excerpt from the introductory section of force-field is shown below for `mm2.fld`, the MM2 force-field.

```
 Allinger MM2(87) Force Field
$Date: 2002/01/24 14:49:02 $ $ $Revision: 1.1 $
 N.L. Allinger, JACS, 99, 8127 (1977); Parameters from NLA (6/87)
 C
 C           Copyright Columbia University 1989
```

```
 C           All rights reserved
 C
 C   Energy Functions and Conversion Factors in Use
 C   ------ --------- --- ---------- ------- -- ---
-1
 0    STR     2      601.99392
 0    BND     2      601.99392
 0    S-B     1      601.99392
 0    TOR     1        2.09200
 0    IMP     1       60.19939
 0    VDW     4        4.18400
 0    ELE     1     1389.50000
 0    HBD     1      418.40000
 0    V14     4        1.00000
 0    FIX     1        4.18400
 0    BCI     2        1.00000
 0    FCH     2        1.00000
 0    SEL     1                 O Original MM2 Parameters
 0    SEL     1                 M Modified MM2 Parameters
 0    SEL     1                 A Added parameters
 0    SEL     2                 1 High quality parameters
 0    SEL     2                 2 Tentative value parameters
 0    SEL     2                 3 Low quality parameters
 0    ALT     1                 a MM2 hydrocarbon params
 0    ALT     2                 b Kroon-Battenburg Lp params
 0    ALT     3                 L Z0 is lithium
```

D.2.1 Equation Specification

The section extending from the `STR` to the `FCH` line above tells the program which equations to use for the various types of interaction and give factors which convert from published force field units to kJ/mol. These data are encoded in lines beginning with `I=0`. Later, in the *Main Parameter* and *Substructure* sections, parameters are specified in whatever units were used in the original publications.

In general, each force-field equation has certain universal constants which do not vary with the atom types in the interaction. It is possible to specify these on the equation-specification line; however, this facility is now used only for BMFF force fields.

Note: The van der Waals factors may be modified when the separation between atoms is near the cutoff.

Stretch

0 None

1 Symmetric (Hooke's law) stretch

2 MM2 asymmetric stretch

3 MM3 symmetric stretch

Bend

0 None

1 Symmetric (Hooke's law) bend

2 MM2 asymmetric bend

3 MM3 asymmetric bend

4 MMFF asymmetric bend

Stretch-Bend

0 None

1 Linear, one constant for both bonds (MM2, MM3)

2 Linear, separate constants for each bone (MMFF)

Torsion

0 None

1 MM2-style cosine torsional potential

2 AMBER-style cosine torsional potential

3 OPLS-style cosine torsional potential with explicit V0

Improper Torsion (Out-of-Plane)

0 None

1 Harmonic out-of-plane (MM2)

2 Improper torsion

3 Improper torsion with explicit V0

4 Harmonic Wilson-angle displacement (MMFF)

Van der Waals

1 7,11-Lennard-Jones equation; Eps, R as defined in AMBER

2 6,12-Lennard Jones Equation; Eps, R as defined in AMBER

3 MM3 Hill Equation; Eps, R

4 MM2 Hill Equation; Eps, R

5 Lennard Jones Equation; Eps, R as defined in OPLS

6 Lennard Jones Equation; Alpha, R, N as defined in CHARMM

7 MMFF buffered 7,14 potential

Coulombic

0 Turned off

1 Constant-dielectric electrostatics

2 Distance-dependent (R) dielectric electrostatics

3 MMFF buffered constant dielectric constant

4 MMFF buffered, distance-dependent dielectric constant

Hydrogen-bonding

0 6,12-Lennard Jones for hydrogen bonds

1 10,12-Lennard Jones hydrogen bonds

2 10,12 Lennard-Jones with angular dependence

Special Treatment of 1–4 Interactions

0 Skip all 1,4 electrostatics and van der Waals

1 Scale 1,4 van der Waals terms by factor given

2 Scale 1,4 Coulombic terms by factor given

3 Scale both 1,4 van der Waals and electrostatics by factor given

4 Do not scale 1,4 nonbonded interactions

Fixed-atom Tethering Potential (When `FXAT` Specified)

0 None

1 Harmonic

D.2.2 Charge Processing

Two new force field descriptors were added as of MacroModel 8.0:

BCI

BCI specifies if MacroModel should expect bond charge increments (BCI's), bond dipoles, or neither from the force field.

0 No such parameters are supplied by the force field. This also includes the Substructure Section of the `.fld` file. To specify charges for force field substructures, you must use explicit partial charges using the characters `' 8'` in the first two columns (see Section D.4 on page 211).

1 Expect BCI's from the force field.

2 Expect bond dipoles from force field.

The default unit for BCI's is the electron. The default unit for bond dipoles is the Debye. 1 Debye = 10E-18 esu*cm = 0.208204 electron*Angstrom If units other than the defaults are supplied by the BMFF force field server or used in the `.fld` file, specify the conversion factor to multiply by to get units of electrons or Debye; otherwise, specify 1.00000.

BCI *requires* that STR specify an equation (not none).

FCH

FCH specifies how MacroModel should handle adding of delocalized formal charge into the charge on a given atom to get the total fixed partial charge on that atom.

0 Do not add delocalized formal charges into the charges on atoms.

1 Use *delocalized* formal charges from the force field server. Delocalization by MacroModel of formal charges supplied by the force field server is not supported.

2 Use delocalized formal charges based on the atom type formal charges in `atom.typ` (or `.fld` if `DEBG 11` is specified).

Conversion factors are not supported for FCH. Formal charge is always taken to be in units of electrons. Any conversion factor is ignored.

FCH *requires* that ELE specify an equation (not none).

For further explanation of how MacroModel handles charges, see Section 2.4 of the *MacroModel User Manual* as well as the BDCO opcode in this manual.

D.2.3 ALT and SEL Specification

The lower parts (ALT, SEL) allow selection among various parameter types (O for original, M for modified, etc.) and among various alternative parameter sets (e.g., Allinger MM2 vs. Osawa MM2').

The SEL (selection) descriptors above allow you to include (or exclude) the lines of the force fields having certain labels. For example, associated with SEL 1 are three symbols O (for Original), M (for Modified), and A (for Added) which denote parameters which are original with the field, modified from the original values or totally new entries. In the SEL lines above, all such parameters are turned on, i.e., to be included into the force field which is read by the modeling programs. If you wish to use only original parameters, delete the lines which select Modified or Added parameters (i.e., SEL 1 M and SEL 1 A). The O, M, and A symbols go with selection switch 1. SEL 2 has associated symbols 1, 2, and 3, signifying high-, medium-, and low-quality parameters. If the SEL 2 2 and SEL 2 3 lines above are removed, the only high quality parameters will be used in the calculation (however, there are no high quality parameters as yet for many common substructures). You can add your own SEL switches starting with SEL 3 and choose any symbols you wish for switching. The symbols you use are the same in the SEL lines above and in the corresponding entries of the force field below. MacroModel allows a total of 5 different SEL's.

The ALT (alternative) descriptors are similar to the SEL descriptors except that they select single possibilities from among various options. We have defined ALT 1 to toggle between the original MM2 parameters (signified by the symbol a) and the Osawa MM2' parameters (signified by the symbol b). By default, a is selected. You may notice that there are seemingly duplicate entries in the force field with slightly different parameters (MM2 and MM2'), these are switched between by selecting the ALT symbol you want.

Figure D.1 on page 196 shows how SEL and ALT options are encoded later, on an interaction-by-interaction basis, in the *Main Parameter* and *Substructure* sections of the force field. Each interaction may have values specified for the various SEL and ALT options. The SEL values should be aligned below the columns labeled 1 2 3 4 5. The ALT number goes below Alt the particular ALT symbol goes below the a. Comments should be used (30 characters maximum) since they are printed out along with the parameters in the output .mmo files containing detailed energy listings. These comments are quite helpful in checking parameters.

Figure D.1.

```
C  Field 1 (Origin):  O = Original MM2, M = Modified Parameter, A = Additional Parmeter
C  Field 2 (Quality): 1 = Final Values, 2 = Tentative Values, 3 = Low Quality Value
C  Field 3
C  Field 4
C  Field 5
C  Alternate 001: a = Original MM2, b = Osawa (Tet, 2769 (83)) MM2' modification of C and H parameters.
C  Alternate 002: a = Original MM2, b = Kroon-Batenburg (JMolStr, 417 (83)) special Lp-H interaction
C                                   to give proper hydrogen bonding geometries and energies.
C  Alternate 003: a = MM2(87) (Allinger, JOC, 5162 (87)), b = Still/Goldsmith (JOC, 951 (87))
C
C                            Interaction Template:
C  XX - XX - XX - XX      0.0000     0.0000     0.0000  0000 0000 0000 0000    1 2 3 4 5 Alt a Comments
C                        Substructure Interaction Template:
C  XX  XX  XX  XX         0.0000     0.0000     0.0000                         1 2 3 4 5 Alt a Comments
C                        Substructure Charge Template:
C        1        2        3        4        5        6        7        8        9       10 1 2 3 4 5 Alt a
C                                   to give proper hydrogen bonding geometries and energies.
C  Alternate 003: a = MM2(87) (Allinger, JOC, 5162 (87)), b = Still/Goldsmith (JOC, 951 (87))
C
C                            Interaction Template:
C  XX - XX - XX - XX      0.0000     0.0000     0.0000  0000 0000 0000 0000    1 2 3 4 5 Alt a Comments
C                        Substructure Interaction Template:
C  XX  XX  XX  XX         0.0000     0.0000     0.0000                         1 2 3 4 5 Alt a Comments
C                        Substructure Charge Template:
C        1        2        3        4        5        6        7        8        9       10 1 2 3 4 5 Alt a
```

D.2.4 Atom Type Equivalencies

It is possible to equivalence several atom types in any given force field by some simple additions in the force field file. This equivalencing allows the use of a special 2-character descriptor to refer to any one of a set of atoms. These 2-character descriptors may be used anywhere in the force field to avoid creation of multiple entries for related interaction types. An example would define `CU` as the various forms of sp2 carbons (`C2`, `CD`, `CE`) and then substitute `CU` whenever the parameters for `C2`, `CD`, and `CE` are all the same. Such a substitution would replace 3 different interaction descriptors with one interaction descriptor. The AMBER force field makes extensive use of such equivalencing.

The atom equivalence should be specified in the force field just before the stretch interactions. The format is first to type `-5` in columns 1 and 2. The next lines contain the atom equivalence data in the Fortran format `(I2,2X,A2,2X,  29(A2,1X))`. Up to 29 atom types can be equivalenced. After all the atom equivalences are specified, end input by putting a `C`, or any letter, in column 2 of the next line; for example:

```
-5
 1 CX  C1 C2 C3
 2 CY  CA CB CC
 3 NP  NA NB
 C
```

This indicates that `CX` is the name of an equivalency class that includes atom types `C1`, `C2`, and `C3`. Thus, if `CX` is specified in an interaction in the force field, then, whenever atom type `C1` or `C2` or `C3` is found in the matching position of the molecule under study, the interactions specified will be used.

Atom equivalencing is not allowed for wild card atom types, e.g., `C0`, `N0`, `O0`, `00`.

D.3 Main Parameter Section

D.3.1 Interactions

There are separate subsections for stretching, bending, torsional, improper torsional, van der Waals and hydrogen bonding parameters. These sections distinguish parameters by the types of atoms and bonds used. These main atom and bond types are listed at the left of each parameter line. Each stretch, bend, and torsional array in the molecule is tested against lines in the force field until a match is found. When a set of atoms and bonds in the molecule defining a stretch, bend or torsion matches those in the parameter line, then the associated parameters from that line of the force field file are used for that interaction. For a match to be found, the atom types and bond types must be the same in the molecule and in the parameter line.

Within the Main Parameter Section of the force-field, lines beginning with I>0 have the following meanings:

1. Constants for stretching interactions
2. Constants for bending interactions
3. Constants stretch-bend interactions
4. Constants for dihedral interactions
5. Constants for improper torsional (or OPB) interactions
6. Van der Waals constants for pairs of atoms
7. Constants for hydrogen bonding interactions

Toward the right in these lines optional atom descriptors may be given. These list additional atom types which must be attached to each of the main atoms at the left of the parameter line for a valid match. Each descriptor consists of a 4-character field (e.g., `0000` or `C200`) specifies two attached optional atoms. `00` is the wild card atom type symbol. Thus `0000` specifies that there are no required attachments to the corresponding main atom (i.e., just match the main atoms and bonds). `C200` says that a `C2` (an sp2 carbon) must be bound to the main atom in question. A `C2C3` indicates that both a `C2` (sp2 carbon) and a `C3` (sp3 carbon) must be bound. Optional atom fields of each force field file entry must not be left blank in the force field file. If no special attachments to any main atoms are required, then use a `0000` for each atom in the interaction. For stretches there 2 optional atom fields (corresponding to the two main atoms of the stretch interaction). For a bend, there are 3 optional atom fields, for a torsion, there are 4 such fields. Consider the torsional entry from a force field file below as an example:

```
                                v1         v2          v3
 4  C3 - C3 - C3 - H1       0.0000     0.0000      0.3500  0000 H1H1 0000 0000
    ^    ^    ^    ^                                       ^    ^    ^    ^
   atm1 atm2 atm3 atm4           Optional atoms for atom: 1    2    3    4
```

The entry above specifies that anytime a `C3 - C3 - C3 - H1` torsional array is found in which two hydrogens (`H1`'s) are attached to main atom 2, then the torsional force constants given (V_1 = 0.0, V_2 = 0.0, and V_3 = 0.35) will be used. Now consider the following torsion force field entry:

```
 4  C3 - C3 - C3 - H1       0.0000     0.0000      0.2670  0000 0000 0000 0000
    ^    ^    ^    ^                                       ^    ^    ^    ^
   atm1 atm2 atm3 atm4           Optional atoms for atom: 1    2    3    4
```

The entry above notes that when any `C3 - C3 - C3 - H1` array is found, then the force constant V_3 =0.267 will be used regardless of the substitution of any atom (including atom 2). This is because all optional atom entries are `0000`. Both of the above lines may appear in a force field because matching is done top down: the first time an acceptable match is found, the parameters from the matching force field line will be used.

For some torsional arrays, it may be desirable to add higher order parameters. This may be done for V_4-V_6 terms by adding an additional line of torsional parameters and beginning the line with a 54 as shown below:

```
 4  C3 - C3 - C3 - H1      0.0000     0.0000     0.2670  0000 00Si 0000 0000
54                         0.1000     0.0000     0.5000
```

The parameters above specify $V_3 = 0.267$, $V_4 = 0.100$, and $V_6 = 0.500$.

Parameters in the file must be ordered from most specific to most general since the program will assign the parameters to interactions the first time a valid match is found between the molecule and the parameter line. Thus in the example of the two torsional interactions above, the more specific one given first must come first in the force field file. That will allow the program to apply the correct v3 term (0.350) if there are two hydrogens (H1) on atom 2—otherwise C3-C3-C3-H1 interactions will use V_3=0.267.

The above paragraphs describe how to specify up to two atoms alpha to (one covalent bond removed from) each atom in the interaction. By adding up to three additional lines of optional atom descriptors, up to three beta atoms (connected to an on-interaction atom via two bonds) may be specified for each alpha atom. Consider the three O-H stretching entries below:

```
 1  O3 - H2                1.0100     5.0000    -1.0000 1C200 0000    Entry 1
                                                       O200 0000
 1  O3 - H2                1.0100     5.0000    -1.0000  C200 0000    Entry 2
 1  O3 - H2                1.0100     5.0000    -1.0000  0000 0000    Entry 3
```

The third entry shown here could be the O-H stretch of a simple saturated alcohol. The second could specify the O-H stretch of, say, a phenol, which has a C2 (an sp2 carbon) connected to the oxygen. The first entry would not match a phenol, since it specifies that the C2 alpha to the oxygen must be bonded to an O2 (sp2 oxygen). This would match a carboxylic acid.

The lines specifying beta atoms have continuation characters immediately preceding the first set of alpha interactions. In the first entry above, the 1 in the string 1C200 tells the program to read a single line of beta atoms next. This character should be replaced by a 2 or a 3 if there are two or three lines of beta atoms. Beta optional atoms appear directly beneath the alpha optional atom to which they are attached. An alpha atom of type 00 may not have beta atoms specified; however, wild-card types for specific atomic numbers, such as C0, may have beta atoms. Within each group of alpha and beta atoms, more specific combinations must be specified first. For example, if it is desired to match interaction atom X, attached to -A-B and also attached to another A, the -A-B combination must be given before (to the left) of the unqualified -A attachment.

When adding new parameters or modifying old ones, it is wise to use a test structure to be sure the new parameters are being found. If they are not, then you may be using an incorrect atom or bond type or a match is occurring from an entry higher up in the force field file for the stretch, bend or torsional parts of the force field. You may easily test for ordering by putting

your entry at the top of the appropriate section (stretch, bend, etc.). You should also rerun the energy test structures supplied to be sure you have not changed other parameters in error. Also be sure to make the appropriate changes in the interaction descriptors at the far right of each entry so that the parameter set will be correctly identified (as we have defined them) as (Column 1) `O` (original), `M` (modified), or `A` (added), and as (Column 2) `1` (high quality parameter), `2` (tentative parameter) or `3` (generalized parameter). We also add a brief comment and the name of (or reference to) the parameter author. You should add the appropriate comments, your initials and location for each entry you add or modify to keep track of the parameters you have added. You will want to be able to add your new parameters to any force fields we supply in the future. Note that force field lines are up to 130 characters so set your editor to display the entire line.

Some excerpts from `mm2.fld` follow, by way of example. Figure D.2 on page 201 exhibits several stretch interactions.

A positive bond moment makes the first atom listed in the bond entry positively charged. In the fourth entry above, the value of –0.32 Debye makes the `C3` negative and the `H1` positive.

In the first line of the stretch section, a special `C3 - C3` interaction is defined by the optional alpha descriptors `O3O3` which require that both the first `C3` and the second `C3` have attached `O3`s, as in ethylene glycol. This is not a standard MM2 stretch and is therefore marked in the `SEL 1` field with an `A`. The `SEL 2` field is marked `1` since the parameter is a high quality one. The second and third lines contain MM2 and MM2' parameters, and are distinguished at the right hand end of the lines in the labeling of the second line by the `ALT` specifications `1 a` and `1 b`.

As described above, if `ALT 1` is given the `a` value in the Introductory Section of the force field, MM2 (not MM2') parameters will be used.

Some bend parameters are shown in Figure D.3 on page 201. The first few parameters shown are original (`O`), high quality (`1`) MM2 parameters. Also, the environmental dependence of the bending angles (methyl vs. methylene vs. methyne) is handled with optional descriptors for the central atom. The last few parameters shown are highly generalized bend interactions about various sorts of central atoms. The use of wild-card atom types (`00`) and bond types (`*`) means that one of these interactions will match nearly any bend having the given central atom. These are given at the end of the bend section, so that more specific interactions will match first. These entries are labeled `A` (added) and `3` (low quality).

A portion of the stretch-bend part of the force-field is shown in Figure D.4 on page 202. Torsions are shown in Figure D.5 on page 202. The same considerations hold for these interaction types as hold for stretches and bends. Of course, not all force fields have all interaction types; for example, the AMBER force fields have no stretch-bend interactions.

Figure D.2.

```
C         Stretching Interactions (STR)                Opt. Descriptor            Parameter Referencing
C         ---------- ------------                     ---- ----------         --------- ----- -----------------------
C                    Bond Length   Constant  Bond Moment Atm1 Atm2             Select    Altrn
C                         (ang)   (mdyn/ang)   (debye)                         1 2 3 4 5   S A Comment
2
1  C3 - C3               1.5150     4.4000     0.0000  O300 O300               A 1             C(-O)-C(-O), WCS (CU)
1  C3 - C3               1.5230     4.4000     0.0000  0000 0000               O 1         1 a C-C
1  C3 - C3               1.5260     4.4000     0.0000  0000 0000               M 1         1 b C-C, MM2', Tet, 2769 (83)
1  C3 - H1               1.1130     4.6000    -0.3200  C100 0000               A 1             H-C(-Cl), WCS (CU)
1  C3 - H1               1.1130     4.6000    -0.7500  N500 0000               A 2             H-C(-N+), WCS (CU)
```

Figure D.3.

```
C         Bending Interactions (BND)                   Opt. Descriptors
C         ------- ------------                         ---- -----------
C                        Angle  Bending Constant        Atm1 Atm2 Atm3
C                        (deg)    (mdyn/rad**2)
2
2  H1 - C3 - H1         109.4700     0.3200             0000 H1H1 0000         O 1             H-C(H2)-H
2  H1 - C3 - H1         109.0000     0.3200             0000 H100 0000         O 1             H-C(H)-H
2  H1 - C3 - H1         109.4000     0.3200             0000 0000 0000         O 1             H-C-H
2  C3 - C3 - C3         109.5000     0.4500             0000 H1H1 0000         O 1             C-C(H2)-C
2  C3 - C3 - C3         109.5100     0.4500             0000 H100 0000         O 1             C-C(H)-C
2  C3 - C3 - C3         109.4700     0.4500             0000 0000 0000         O 1             C-C-C
2  C3 - C3 - H1         110.0000     0.3600             0000 H1H1 0000         O 1             C-C(H2)-H
..
2  00 * C3 * 00         110.0000     0.4500             0000 0000 0000         A 3             -C(sp3)-, WCS (CU)
2  00 * C2 * 00         120.0000     0.4500             0000 0000 0000         A 3             -C(sp2)-, WCS (CU)
2  00 * C1 * 00         180.0000     0.4000             0000 0000 0000         A 3             -C(sp)-, WCS (CU)
```

Figure D.4.

```
 C         Stretch Bend Interactions (S-B)              Opt. Descriptor
 C         ------- ---- ------------                    ---- ----------
 C                     Str Bend Const                    Atm1 Atm2 Atm3
-2
 3  H1 * C1 * 00           0.0900                       0000 0000 0000         O 1             H-C(sp)*
 3  00 * C1 * 00           0.1200                       0000 0000 0000         O 1             *C(sp)*
 3  H1 * C2 * 00           0.0900                       0000 0000 0000         O 1             H-C(sp2)*
 3  H1 * C2 . 00           0.0900                       0000 0000 0000         O 1             H-C(sp2).
 3  00 * C2 * 00           0.1200                       0000 0000 0000         O 1             *C(sp2)*
```

Figure D.5.

```
 C         Torsional  Interactions (TOR)                  Opt. Descriptor
 C         ---------  ------------                       ---- ----------
 C                           V1          V2          V3   Atm1 Atm2 Atm3 Atm4
 C                          (V1, V2 & V3 in kcal/mole)
-2
 4  H1 - C3 - C3 - H1      0.0000      0.0000      0.2370  0000 0000 0000 0000    O 1         1 a H-C-C-H
 4  H1 - C3 - C3 - H1      0.0000      0.0000      0.3500  0000 0000 0000 0000    M 1         1 b H-C-C-H MM2', Tet, 2769 (83)
 4  C3 - C3 - C3 - H1      0.0000      0.0000      0.2670  0000 0000 0000 0000    O 1             C-C-C-H
 4  C3 - C3 - C3 - C3      0.2000      0.2700      0.0930  0000 0000 0000 0000    O 1         1 a C-C-C-C
```

Low quality (3) torsional parameters are the most likely to give incorrect results in molecular mechanics. If generalized torsions are used in a calculation, then quantitative results should not be expected and new torsional parameters should be devised to fit the molecule on the basis of experimental (or ab initio quantum-mechanical) data. Minimally, the results of a calculation with low quality torsion should be compared with experimental data to determine how poor the calculation is. See the *MacroModel Technical Manual* for some comments regarding parameter derivation.

Figure D.6 on page 204 shows an excerpt from the section of the force-field giving out-of-plane bending parameters. When the these interactions are handled using improper torsions, the atoms are listed with the central atom (e.g., the carbon of a carbonyl group) first in the entry; that is, if the atoms are listed in this section in the sequence `ABCD`, the dihedral angle calculated will really be `BACD`. When these interactions are being handled using Wilson angles, the two "anchor" atoms are given first, then the "apex" atom, then the "end" atom (Wilson, E. B.; Decius, J. C.; and Cross, P. C. *Molecular Vibrations*; Dover Publications; New York, NY, 1955; p. 59).

A section of the van der Waals portion of the force field is shown in Figure D.7 on page 204. This section gives single-atom van der Waals parameters which will be combined using some combing rule. The parameters given are r_0 and ε.

Note the use of optional atoms to distinguish MM2 atom types and the `Lp` descriptor at the right to indicate which atoms get lone pairs; for example, an `O3` not bound to `P`, `S` or `C`(sp2) gets a lone pair. MacroModel does not use lone pairs on enol, ether or ester oxygens this gives better results with hydrogen bonding. In the past it was not possible to specify beta atoms for this portion of the force field, but this is now allowed.

Formal charges for atom types are obtained from the `atom.typ` file, rather than from this section of the force-field file.

Following the van der Waals parameters, there are two more parts of the Main Interaction Section of the force field. Figure D.8 on page 205 shows some of the special van der Waals interactions. These encode overrides of the normal van der Waals combining rules for atom-type pairs. Figure D.9 on page 205 shows some hydrogen-bonding parameters. Even when there is no special functional form invoked for hydrogen bonding, the parameters for whatever standard nonbonded potential is being used is derived from this section for atoms that can hydrogen bond.

Figure D.6.

```
 C         Out of Plane Bending                          Opt. Descriptor
 C         --- -- ----- -------                          ---- ----------
 C                         Angle      Const.             Atm1 Atm2 Atm3 Atm4
-2
 5  C2 * O2 * 00 * 00      0.0000     0.8000             0000 0000 0000 0000    O 1
 5  C2 * 00 * 00 * 00      0.0000     0.0500             0000 0000 0000 0000    O 1
 5  CP * 00 * 00 * 00      0.0000     0.0500             0000 0000 0000 0000    A 2            C+, TL 1703(84)
 5  CR * C2 * 00 * 00      0.0000     0.8000             0000 0000 0000 0000    A 2            C*(Csp2), WCS (CU)
```

Figure D.7.

```
 C         Van der Waals Interactions                          Opt. Descriptor
 C         --- --- ----- ------------                                ---- --
 C       Radius      Eps       Offset     Charge                     Atm1 Lp
 C        (ang)   (kcal/mole)   (ang)
-6
  C1     1.9400     0.0440     0.0000                                0000    O 1
  C2     1.9400     0.0440     0.0000                                O200    O 1
  C2     1.9400     0.0440     0.0000                                0000    O 1
  C3     1.9000     0.0440     0.0000                                0000    O 1         1 a
  C3     2.0000     0.0440     0.0000                                0000    M 1         1 b Osawa MM2', Tet, 2769 (83)
```

Figure D.8.

```
 C         Special Van der Waals Interactions           Opt. Descriptor
 C         ------- --- --- ----- ------------           ---- ----------
 C                         Radius      Eps              Atm1 Atm2
 C                         (ang)   (kcal/mole)
-2
 6  C3   H1                3.3400     0.0460            0000 0000            O 1             Special C3-H1 part of MM2
 6  H2   Lp                2.4000     0.0240            0000 0000            O 1         2 a
 6  H2   Lp                2.4900     0.0187            0000 0000            A 2         2 b LKB (JMolStr, 417 (83)) HBonding
 6  H3   Lp                2.5250     0.0233            0000 0000            O 1         2 a
 6  H3   Lp                2.4900     0.0187            0000 0000            A 2         2 b LKB (JMolStr, 417 (83)) HBonding
```

Figure D.9.

```
 C      Hydrogen Bond Interactions (function and params from AMBER
 C              modified for Lp-containing oxygens)
 C      ----------------------------------------------------------
 C Acceptor Donor           C/100     D/100              Atm1 Atm2 Atm3 Atm4
-2
 7  H1   O3               90.0000    40.0000             S100 0000
 7  H3   OM               72.0000    24.0000             0000 0000
 7  H3   O3               30.0000    12.5000             0000 0000
 7  H3   O2               42.5000    10.0000             0000 0000
 7  H3   F0               75.0000    24.0000             0000 0000
```

D.3.2 Adding New Parameters

New parameters generally come from one of three possible sources: new published work, quantum-mechanical calculations and educated guesses based on analogy to systems similar to that being parameterized. Some examples of parameters based on quantum-mechanical calculations are given in the *MacroModel Technical Manual*.

New parameters from MM2, MM3, and OPLSA publications may be added directly to the force field files without modification. If specific charges are desired, you may convert the desired charges to a dipole entry in the stretch section of the field using the following formula:

$$\mu = L_0 * Q \ / \ 0.2082$$

Here, μ is the dipole moment in debyes, L_0 is the natural length of the bond (in Angstroms, Å) given in the force field and Q is the desired charge (+ on one end and – on the other).

A positive dipole moment means that the first main atom in the force field entry will have the positive charge and the second atom will get an equal but negative charge. For the opposite polarity, use a negative dipole moment. Again, test by doing an energy calculation on a structure with MacroModel via Maestro, and then using the Electrostatics panel to display the resulting partial charges. The Electrostatics panel can be displayed by choosing Force-Field Viewer from Maestro's Analysis menu, then clicking the Electrostatics button.

For stretches and bends, new values may be assigned to fit experimental data as far as bond lengths and angles are concerned. Such data commonly comes from high quality x-ray crystal structures. Force constants can be approximated by analogy with similar structures in the force field or from infrared stretching frequencies:

$$K(\text{stretch-MM2,MM3}) = 5.3\text{x}10^{-7}\ \nu^2\ M_1\ M_2\ /\ (M_1+M_2)$$

$$K(\text{stretch-AMBER}) = 3.0\text{x}10^{-5}\ \nu^2\ M_1\ M_2\ /\ (M_1+M_2)$$

where ν is the IR frequency in wave numbers (cm^{-1}), and M_1 and M_2 are the atomic masses of the atoms involved in the stretch. Alternatively, you may transfer force constants from one field to another. AMBER stretching force constants are approximately 60 times those of MM2 or MM3.

For bending, infrared frequencies may also be used (if they are available for the scissoring mode) from the following approximate expression:

$$K(\text{bend-MM2,MM3}) = 3.0\text{x}10^{-7}\ \nu^2\ M_1\ M_3\ /\ (M_1+M_3)$$

$$K(\text{bend-AMBER}) = 3.4\text{x}10^{-5}\ \nu^2\ M_1\ M_3\ /\ (M_1+M_3)$$

where ν is the IR frequency in wave numbers (cm^{-1}), and M_1 and M_3 are the atomic masses of the first and third atoms involved in the bend. AMBER bending constants are approximately 120 times those given in MM2 or MM3 for the same bending array.

While stretch (and sometimes bend) parameters may be moved from one force field to another without creating large errors, torsional parameters are highly force field dependent for the same array of atoms. Consequently, new torsions must be parameterized in the context of the field to be used. Often, data from one well-parameterized force field (e.g., MM2 or MM3) can be used as the basis for parameterization of another force field.

Nonbonded parameters for unusual atoms may be approximated according to the following equation [50, 51]. The van der Waals ε is given by:

$$\varepsilon = 1.4154\ \alpha^2 r^{-6}\ (\alpha/N_{el})^{-1/2}$$

where r is the covalent radius (ionic radius for ions) in Angstroms, α is its polarizability ($Å^3$) and N_{el} is its effective number of electrons for the atom (generally equal to the actual number of electrons for first and second row elements).

While these protocols work well, it is easy to corrupt the force field by careless addition of new parameters or modifications of old ones. If you make any additions to the force fields, be sure and test that the additions you make do not change the energetic results of a adequately wide variety of test structures. In particular, unless it is your intent to have your new parameters alter even the structures which are well parameterized with the force-field as we supply it, you should check to make sure that the energy tests described in Appendix A still work properly.

D.3.3 Special Notes for AMBER

AMBER94 does not differ from the published force-field in any significant way.

The MacroModel implementation of AMBER* is identical to original AMBER with the exception of additional parameters which we have added. We supply the field by default with a constant dielectric treatment, the united-atom AMBER charge set and Kollman's 6,12-Lennard Jones hydrogen bonding treatment. Other options (e.g., distance dependent dielectric) may be set by modifying the first section of the AMBER force field file or via button selections from within MacroModel.

The AMBER force field we supply has a special substructure notation, which allows a single residue substructure to match both united atom and all atom structures and assign the appropriate charges as well as other parameters. Although the linear notation gives only the united atom description of the molecule, the program will interpret the notation both as an actual united atom representation and as the all atom representation with the appropriate number of explicit hydrogens. The charges given for the residues are in the united atom representation for

both the united atom and the all atom force field parameter sets. Thus, for the all atom charge set, the charges given are for the heavy atoms plus any attached hydrogens. To get the actual complete all atom charge set, the program uses dipoles from the main section of the field (mainly for heteroatom-H's) or from the substructure (mainly for C-H's).

By using a combination of bond dipoles and formally united atom charges, MacroModel reproduces both AMBER field charges exactly and also provides reasonable charges for substituted or modified residues. As such, it allows the use either charge set (from the All atom or United atom paper) and, independently, hydrogens on carbons or united atoms. As we supply the field, the charges correspond to those given in the older AMBER paper (i.e., the united atom paper), but you may switch to charges from the newer paper (i.e., the all- atom paper) by modifying the ALT 1 line at the top of the force field file.

You will notice that many of the parameters for some of the more complex amino acid sidechains (phenol, indole, imidazole, etc.) are given separately from the actual residue itself in the substructure section of the force field file. This allows any such substructure to be given the same parameter set as used in the actual amino acids. Thus indole itself gets the same parameter set as the indole of tryptophan. Such substructures are labeled as "C" (for Continue, see below) which means that the atoms matching the substructure may also be matched against substructures further down in the force field (i.e., where the actual residue might be found). The residues themselves are labeled "S" (for Stop). This means that once atoms are matched (as a complete residue), then they are removed from candidacy for matching with substructures which come below them in the field. This distinction speeds substructure matching since each such match of a "S" substructure removes atoms which must be tested from subsequent matching.

Amber Torsional Specification

There are three differences between the MacroModel and the AMBER torsional parameters. These differences do not affect the actual functional forms used by the program, but only the way in which parameters are specified.

First, AMBER uses a different entry for each 2-fold (n=2) and each 3-fold (n=3) barrier; MacroModel uses a single entry with both 2-fold (V_2) and 3-fold (V_3) barriers included. Second, AMBER uses only positive torsional potentials but uses an angular offset (γ) to adjust the position of minima and maxima. For γ=0° or γ=180°, this is best accomplished in MacroModel through the use of positive and negative Vn values, as follows.

AMBER n=3, γ= 0°: use $V_3/2$ directly

AMBER n=3, γ=180°: use $-V_3/2$ for $V_3/2$

AMBER n=2, γ= 0: use $-V_2/2$ for $V_2/2$

AMBER n=2, γ=180: use $V_2/2$ directly

Some AMBER publications give torsional parameters as $V_n/2$ and some as V_n. In MacroModel force-field files, always use the $V_n/2$ value.

We also implement AMBER torsional offsets explicitly for arbitrary offsets; however, the implementation is not as efficient as that described in the preceding two paragraphs. Thus, for example, you may, in principle, specify an n=2 potential using a 180° offset by means of a positive value of $V_n/2$ and an explicit value of 180°, but you are advised instead to use the formulation given above. On the other hand, for offsets equal to neither 0 nor 180°, there is really no choice but to use an explicit specification.

To specify offsets explicitly for V_1, V_2, and V_3, use a continuation force-field line beginning with `64` and place the offsets in the same columns as the V values for the main torsion line. Similarly, to specify offsets for V_4, V_5, and V_6, use a continuation line beginning with `74` and insert the offsets in the same location. (Recall that V_4, V_5, and V_6 themselves use a continuation line beginning with `54`.) An offset specified as 0 is taken to mean "no offset specified," so that if you wish to use a V_2 potential with a zero offset and a V_3 potential with a 31° offset, use a negative value of $V_2/2$, as described above, on the first torsion line for this interaction; place the positive value of $V_3/2$ on the same line. On a continuation line beginning with `64`, indicate a V_2 offset of zero and a V_3 offset of 31°.

AMBER publications give total barriers for generalized torsional arrays—arrays with wild card atoms (the `X` atom type in AMBER publications) at positions 1 and 4. MacroModel, however, uses the MM2 method of storing only the component torsional barriers except in the substructure part of the field. Thus only generalized AMBER torsional barriers should be divided by the multiplicity of the torsional array where the multiplicity if equal to the number of substituents on the 2nd atom (excluding the 3rd atom) multiplied times the number of substituents on the 3rd atom (excluding the 2nd atom) in a torsional array of four atoms: 1-2-3-4. For the central bond of hexamethyl ethane for example, the multiplicity would be 3*3 or 9. For the peptide linkage, the multiplicity would be 2*2 or 4. Note that the multiplicity is computed based on the actual number of substituents so the multiplicity of a pair of united atom methylenes, e.g., `x - CB - CB - x` (MacroModel) or `X - C2 - C2 - X` (AMBER), would be only one since only single substituents (`X`) are attached to the methylene carbons. If you are confused, compare some of the entries in the MacroModel AMBER force field with those published in the AMBER papers. Remember that if a torsional array is described in AMBER publications without wild card `X` atoms, then the barriers are simply adjusted by sign as described above, but not divided by multiplicity.

Regarding AMBER-style generalized torsions, MacroModel follows the AMBER protocol of not including the multiplicity for generalized torsions in substructures. only. Thus the substructure torsional barriers are the same as published in the AMBER papers if they are in the force

field file in the form of `00` *nn nn* `00`, where *nn* is some atom number. The beginning and ending `00`'s are wildcard atom types.

D.3.4 Special Notes for MM2 and MM3

The MM2 field we supply is similar to but not identical with that provided by Allinger in 1987. The MM3 parameter set is the 1991 version. The stretch, bend, torsion, and off-diagonal (e.g., stretch-bend) equations are identical with those of the authentic fields. Included in the MacroModel fields are a number of additional parameters to handle substructures other than those parameterized by Allinger. Such parameters are labeled at the right end of the parameter line as `A` for Addition. We have followed Allinger's protocol of distinguishing parameter quality by noting his final values as `1` and other, less reliable ones, as `2` or `3'` (also at the right of each parameter line). When you report the results of a MacroModel MM2 or MM3 calculation, then you should report what nonstandard, additional parameters (if any) are used.

More significant departures involve charges. MacroModel uses the partial charge (optional in MM2) treatment of electrostatics instead of the MM2/MM3 standard dipole-dipole electrostatics. Partial charges for the actual MM2 or MM3 parameters come from the classical definition of a bond dipole and thus correspond to the MM2 or MM3 dipoles exactly. Electrostatic energies will however be different from those given by the authentic Allinger programs and will differ the most when the charged/dipolar groupings are close together. As we provide them, MM2 and MM3 force fields use distance-dependent dielectric electrostatics.

The MacroModel implementation of MM2/MM3 includes approximate parameters for hydrogen bonds which were adjusted to mimic AMBER hydrogen bonding potentials. Part of the hydrogen bonding implementation included the removal of lone pairs from ester and aryl ether oxygens which gave too high a hydrogen bonding energy—this removal of lone pairs from ether oxygens bound to sp2 centers is another departure from the official MM2.

Whereas MM2 and MM3 use a SCF pi calculation to obtain parameters for conjugated systems, MacroModel uses specific torsional parameters from the force field for conjugated dienes, enones, arenes, etc. Where available, these torsional parameters were developed to fit data from Allinger's papers or from MM3. Otherwise, they were developed to fit ab initio data. Aromatic ring parameters are given in the special substructure section of the field.

A final relatively insignificant difference is the way out-of-plane bending is handled. In MM2 and MM3, an out-of-plane distance is computed whereas in MacroModel we use an improper torsion with a barrier which mimics the MM2/3 out of plane energy for moderate excursions from planarity.

D.4 Substructure Section

D.4.1 Use of Substructures

The last section of the force field file lists special substructures. A force-field substructure is a contiguously connected chemical entity, such as an amino-acid residue or an aromatic nucleus. The connectivity is given by a simple linear notation. For each substructure, parameters are given that override any parameters obtained for the corresponding interactions from the Main Parameter Section of the force-field. These parameters may lie wholly within the substructure or they may include off-substructure atoms. Single-atom van der Waals parameters may not be specified within substructures and charges may be specified either by means of bond dipoles, as in the Main Parameter Section, or else by means of explicit charges. In contrast to the Main Parameter Section of the force-field, parameters for special substructures must be entered with the most specific parameters last, rather than first; in other words, the last parameter match found is the one that is used. This is also true for the substructure matching process: if a given portion of the molecule under study is matched by several force-field substructures, the substructure that appears last in the force field is normally used. For substructure matching, but not for parameter matching within substructures, this behavior may be overridden in the force-field specification of the substructure.

Within the Substructure Section of the force-field, lines with `I>0` or beginning with character data have the following meanings; this list augments the values in the Main Parameter Section. Recall that `I` is a two-character field; single values shown must be right-aligned in this field.

- `8` Constants for charges in special substructures.
- `9` Substructure linear notation (see section below).
- `C` Data lines beginning with a blank followed by a `C` or `S` mean “continue” or “stop” in the substructure-matching process.
- `aT` Data lines beginning with two ASCII characters (e.g., aT or aR) indicate a geometrical constraint line, as described in the Geometry Dependent Parameters Section.
- a*n* Data lines beginning with a character followed by an integer encode interactions subjected to geometry dependence, as described in the Geometry Dependent Parameters Section. The letter refers to the corresponding geometric constraint line and the integer specifies the interaction type (e.g., `1` for stretch) as usual.

Note: Single-atom van der Waals parameters may not appear in substructures.

Note: Partial charge line(s) in force field substructures (specified by the `8` flag in the first two columns) *must* be the last line(s) in the substructure, that is, lines for other interaction

types for that substructure, if they exist, must precede the partial charge parameter line(s).

D.4.2 Substructure Linear Notation

In addition to the normal one- to four-atom descriptors of stretch, bend, and torsional substructures used in molecular mechanics, MacroModel provides for description of more complex substructures. These are the "special substructures" given at the end of a MacroModel force field files. The description of the atomic connectivity is given by a simple linear notation which resembles SMILES/SMARTS notation. Atoms and bonds are described as in normal interactions but up to 50 atoms (130 characters maximum) may comprise a substructure, and complicated connectivity, including wild-card and optional atoms, may be specified.

Structures are described as a list of connected atoms and bonds; all atoms and bonds in the structure must be explicitly included. Ethyl vinyl ether would be:

```
C2=C2-O3-C3-C3
```

The atom types are normal MacroModel atom types: `C2` is sp^2 carbon, `C3` is sp^3 carbon, and `O3` is non-carbonyl oxygen.

Simple generalized atom descriptors may also be used; then the formula above would be:

```
C=C-O-C-C
```

You may also use the equivalenced atom types to represent many different types of atoms. Generalized bond descriptors may also be used; for example, an asterisk (`*`) stands for a bond of any order:

```
C*C-O-C-C
```

This could represent diethyl ether, ethyl vinyl ether or ethoxy acetylene.

D.4.2.1 Ring Closure

Ring-forming bonds are added as a bond type followed by a number which specifies which atoms are bound together. Cyclopentane is:

```
C-C-C-C-C-1
```

Thus the last (fifth) atom is bound to the first (`-1`); this signifies a single bond to the first atom in the substructure. Ring closure bonds can go only to atoms earlier in the substructure. Norbornane would be represented:

```
C-C-C-C(-C-1)-C-C-1
```

Note the use of parentheses to identify atoms off the linear backbone as represented. Quinoline could be:

```
N=C-C=C-C=C(-1)-C=C-C=C-5
```

For the quinoline above to match, it would have to have the connectivity and bonding exactly as drawn above. Thus the user would always have to draw the correct resonance structure for the parameters to be found. As an alternative, we generally specify the atom type more exactly to designate the atomic hybridization and use wild card bonds for all the connectivity. Thus any resonance form of quinoline would match the following:

```
N2*C2*C2*C2*C2*C2(*1)*C2*C2*C2*C2*5
```

D.4.2.2 Chain Branching

As a further example of the use of parentheses, methylcyclobutane could have any of the following linear notations:

```
C(-C)-C-C-C-1
C-C-C-C(-C)-1
C-C-C-C(-1)-C
C-C(-C)-C-C-1
```

Parentheses may be nested. Isopropylbenzene could be:

```
C=C(-C(-C)-C)-C=C-C=C-1
```

Threonine could be:

```
N-C(-C(-C)-O-H)-C=O
```

D.4.2.3 Optional Atoms

Optional atoms may also be specified within the linear notation so that the substructure will match whether or not the optional atom is found. Optional atoms with associated bonds are put in square brackets ([ ]). In the substructure below, serine would be identified whether or not the optional hydrogens were present in the structure:

```
N[(-H)]-C(-C-O[-H])-C=O
```

Parameters are assigned for optional atoms if the atoms are explicitly present in the molecule. If the optional atoms are absent, the constants involving those atoms will not be used in reassigning the interaction constants previously assigned by the first part of the force field (1-4 atom interactions). Another way to accomplish the same end is to omit the optional atoms but to include the parameters involving the hydrogens as off-substructure interactions (see below).

The atoms in the substructure are referred to by number when parameters are assigned. An atom is simply referred to by its position in the linear notation; for example, the serine nitrogen would be number one and the carbonyl oxygen would be number 8.

D.4.3 Substructure Charge Processing

You can continue to specify explicit partial charges in force field substructures by specifying `8` in the first two columns of a line containing such partial charges as in previous versions of MacroModel. However, internal processing of these charges was changed in version 8.0. Whereas MacroModel used to directly "paste" such partial charges onto the system being simulated, it now proceeds as follows:

1. Formal charges from `atom.typ` are assigned to the atoms in the force field substructure (`DEBG 11` causes formal charges to be assigned from the `.fld` file).
2. Such formal charges are delocalized over other atoms in that force field substructure if appropriate.
3. The delocalized formal charges are subtracted from the user specified partial charges.
4. These modified partial charges are decomposed into bond charge increments.

Internally, all charge processing to determine partial charges on the system being simulated is done using bond charge increments and delocalized formal charges. For detailed output to the `.log` file of this process, specify `DEBG 128` at the top of your `.com` file. Note that if wildcard atom types (e.g., N0, O0, 00) or generalized atom types (e.g., C, O) are used in the substructure linear notation, such atoms will be assigned formal charges of 0.0, and formal charge delocalization will not be allowed over those atoms in the delocalized formal charge calculation for that force field substructure. Additionally, formal charge delocalization will not be done over wildcard bond types, denoted by "*" in the substructure linear notation. If equivalenced atom types are used in the force field substructure, the formal charge properties will be taken to be those for the first MacroModel atom type in the list of atom types for that particular equivalenced atom type. For example, given the line

```
1  CS  CA CB CC
```

the formal charge properties for an atom of type CS in a force field substructure will be those which are specified for the actual MacroModel atom type CA.

D.4.4 Substructure Examples

The following example shows a substructure specification for a cyclopropane nucleus. The `-3` and `-2` that appear alone on lines specify the format for the reading of subsequent lines. The `C`, `9`, `1`, `2`, etc. introducing other lines specify the nature of the information to be read. The `C` specifies that the title of a substructure is to be read, with the default ("continue") matching

behavior; an S ("stop") in place of `C` would prevent the matched atoms in the structure being studied from being candidates for further substructure matches. In other words, `S` would override the "use last substructure match" default behavior for this particular substructure. `9` introduces a linear substructure specification within the `-3` format block and `1`, `2`, etc. introduce parameter specifications as follows, within the `-2` format block.

```
-3
 C  Cyclopropanoid
 9  C3-C3-00-1
-2
 1   1   2                 1.5010     4.4000     0.0000
 1   1  C2                 1.4670     4.4000    -0.1500
 1   1  C3                 1.5050     4.4000     0.1500
 1   1  H1                 1.0860     4.6000     0.0000
 2   1   2   3            60.0000     0.4500
 2   1   2  H1           119.7000     0.3600
 2   1   2  C3           118.2000     0.6000
 2   1   2  C2           118.2000     0.6000
 2  H1   1  H1           120.6000     0.3200
 2  C3   1  H1           123.5000     0.3600
 2  C3   1  C3           120.0000     0.4500
 2   1  C3  C3           112.4000     0.4500
 4   1   2  C3  H1         0.0000     0.0000     0.1200
 4   1   2  C3  C3         0.3000     0.0000     0.4500
 4  C3   1  C3  H1         0.0000     0.0000     0.6900
 4  H1   1  C3  H1         0.0000     0.0000     0.6900
 4  C3   1  C3  C3         0.0000     0.0000     0.0000
 4  H1   1  C3  C3         0.0000     0.0000     0.2670
```

Substructure interaction lines can specify optional connected atoms, but if they do, then non-wildcard atom-types in the optional-atom list will be matched only against atoms not in the substructure. In this context, a wildcard atom-type is either `00`, which matches any atom type, or else one of the wildcard types for an element; for example, `C0` for any carbon. The numbers in the interaction proper (such as `1 2` in the first stretch shown) correspond to the numbers in the linear substructure notation. Atom types, when shown, correspond to matching atoms off the substructure.

The following example shows a substructure for benzenoids, giving benzene-like bond lengths, angles, and force constants. Wild-card bond types are used to allow matching of any resonance form. Since the substructure is matched a total of 12 different ways, all aromatic bonds will be parameterized by this substructure (with length = 1.39 Å, force constant = 8.0667), etc. The torsion is generalized (the first and last entries are `00`, the wild-card atom type), so it represents the total barrier (recall that this handling is special for generalized torsions in substructures). The barriers used by the individual components would thus be V_2 = 36.8/4 kcal/mol.

```
-3
```

```
 C  Benzenoids (Beckhaus, ChemBer, 116, 86 (1983))
 9  C2*C2*AA*AA*AA*AA*1
-2
 1   1   2                 1.3900     8.0667     0.0000
 2   1   2   3           120.0000     0.4300
 4  00   1   2  00         0.0000    36.8000     0.0000
```

The usual torsional parameter handling in substructures is that all parameters (V_1, V_2, and V_3) for a matched torsional interaction are overwritten by the torsional parameters from a torsional substructure parameter line. It is possible to overwrite only specific components (e.g., only the V_2 component) by using the special parameter entry 99.9999 for the other components; this prevents overwriting. For example, the torsional parameter line below would replace only the V_2 barrier (with 2.5 kcal/mol) of an existing torsional interaction.

```
 4  00   2   3  00        99.9999     2.5000    99.9999
```

Note again that the V_2 barrier specified here is the total barrier, because of the use of wild-card atom types.

Figure D.10 on page 217 shows how a specific set of charges (perhaps derived from ab initio quantum-mechanical studies) can be specified directly in a substructure. The purpose of the substructure above is to load the partial charges given in the line beginning `8`. The charges are placed on the substructure atoms in the order in which the atoms appear in the substructure notation. When using partial charges directly instead of bond dipoles, be sure that the total charge of the substructure sums to 0.0 for neutral substructures. Also check the final charge set using Maestro (the Electrostatics panel) to test your charge data.

D.4.5 Geometry-dependent Parameters

It is occasionally desirable to make the parameters used for certain degrees of freedom dependent upon the geometry of some other degree of freedom. We allow this, in a limited sense. First, the degree of freedom which dictates the parameters must be configurational; that is, we do not provide a mechanism for “on-the-fly” reparameterization of atoms, but only for parameterizing some degree of freedom based on the initial value of some other. Second, we provide three specific forms of this dependence. First of all, a parameter for an interaction can depend on a torsion value, and secondly a parameter for an interaction can depend on whether the interaction is within a ring; finally, a parameter can depend on the initial value of an angle. The following examples illustrate several cases.

Figure D.10.

```
-3
 C  Imidazolium (AMBER HIP)
 9  CU-N4=CU-N2-CU=1
-2
 1   1   2                 1.3830   424.5000
...
 8   0.1160   0.2480   0.2720   0.2480   0.1160                                    O 1         1 a JCC, 230 (86)
 8   0.3530  -0.2125   0.7190  -0.2125   0.3530                                    O 1         1 b JACS, 765 (84)
```

```
-3
 C  Pyranose sugar
 9  O3-C3-O3-C3-C3-C3-C3-C3-2
-2
aT   1   2   3   4       180.0000    60.0000
 1   1   2                 1.3000     5.0000     0.0000
a1   1   2                 1.4000     5.0000     0.0000
```

This shows how parameters may be selected depending on the value of a torsion. Here, we wish to load one exocyclic `O3-C3` bond length (1.3 Å) for an a-anomeric (axial) sugar, and a different bond length (1.4 Å) for a b-anomeric (equatorial) sugar. The entry beginning `aT` defines a geometrical constraint which is to be tested just before the parameters are loaded. The `T` defines the constraint as a torsion. The `a` could be any character (a-z, A-Z) and is used to associate particular subsequent interaction lines with the constraint. The `1 2 3 4` torsion angle, because it is preceded with an `a`, will be subjected to the `aT` specification. The constraint test will passed if the torsion angle of the equivalent atoms in the molecule is in the range 180° 60°. For β-anomeric sugars, this torsion angle would be approximately 180° (anti). In the example, the 1-2 bond (the first `O3-C3` in the substructure) will be given a natural length of 1.3 Å by default, but if the torsional constraint labeled "a" is met, it will be given the value 1.4 Angstrom. This is specified by prefixing the interaction line containing this bond-stretch parameter value by the letter "a." The the more specific (here `a1'`) interaction line must come second in the list of substructure force field entries, since in these entries the last match found overrides the others. If the default value were given last, it would always be used.

The following excerpt exhibits a more complex torsional dependence of several parameters, and also a dependence on whether several atoms are present in a ring. Here, parameters may be dependent upon three different torsions, given in the lines beginning with `uT`, `vT`, and `wT`, or on whether the atoms in the molecule corresponding to substructure atoms 2 and 10 are in a ring of size eight or smaller, as specified in the line beginning with `rR`.

```
-3
 C  Diels-Alder TS (activ), Houk, JACS, 4796 (1992)
 9  O=C2-C2=C2[(-C3)].C2[(-H1)]=C2-C2=C2[(-H1)].3
-2
uT   2   3   4   5       180.0000    60.0000
vT   8   9  10  11       180.0000    60.0000
wT   9   8   6   7       180.0000    60.0000
rR   2  10                 8.0000
...
 1   4  H1                 1.0712     4.6000
 1   4  H1                 1.0736     4.6000             C300
u1   4  H1                 1.0712     4.6000             C300
...
 1   6   7                 1.1720     4.6000
w1   6   7                 1.0743     4.6000
```

```
...
 1  10  11                 1.1683     4.6000
v1  10  11                 1.0740     4.6000
...
 2   2   3  10            88.969      0.0600
r2   2   3  10            88.969      0.0000
 2  H1   3  10            88.563      0.0500
r2  H1   3  10            88.563      0.0000
```

The equilibrium bond length between substructure atom 4 and an off-substructure `H1` is set to 1.0712 Angstrom by default. If the optional `C3` is present on atom 4, it is given the value 1.0736 Angstrom, unless the 2-3-4-5 torsion falls between 120 and 240. In this situation, the `4-H1` bond-length is set back to the default value. The two other torsional dependencies shown are straightforward.

The lines beginning with `r2` specify a test against the ring constraint. The `2-3-10` bending force-constant is set to 0.06 by default, but is set to zero instead if atoms 2 and 10 fall within a ring of size 8 or less. This condition will be met in the internal Diels-Alder transition state. The `H1-3-10` bending force-constant (where `H1` is an off-substructure hydrogen) is subject to a similar test.

The same mechanism can be used to specify the angular dependence of a parameter. The letter `A` in the second column of a force-field substructure line denotes such a dependence. The first real-number field specifies a central value, and the second specifies a range about it. This mechanism can be used, for example, to set equilibrium bond angles in a trigonal bipyramidal complex to 90˚, 120˚ or 180˚, depending on whether each ligand pair is axial-equatorial, equatorial-equatorial or axial-axial. Usually, however, one uses the points-on-a-sphere approach (see `VDWB`) to model such systems.

The BMFF Protocol

E.1 Purpose

BMFF (BatchMin Force Field) is an interprocess communication protocol which allows MacroModel to obtain force-field interactions from an external process. In this context, MacroModel is the client and the external process is the server.

The purpose of this appendix is to provide a functional description of the protocol, rather than to define the underlying implementation details. The current implementation is based on pipes. There is no requirement at present that the server be re-entrant; that is, at present, each client process starts up and terminates its own server process.

In the current implementation, the named pipes for communication between MacroModel and the server process are set up in:

- The directory named in the environment variable `FIFO_LOCATION`, if this variable exists;
- Otherwise, `/tmp`, if this directory exists;
- Otherwise, the local directory.

Note: On SGI systems, named pipes are very slow on NFS-mounted filesystems. Use an explicit `FIFO_LOCATION` in the environment if the above hierarchy does not result in the use of a locally mounted disk for the location of the named pipes.

E.2 Overview

The MacroModel force field continues to be specified by means of the `FFLD` command in the command file. In the past (i.e., prior to the implementation of BMFF), the first argument of this command was limited to the values 1 through 5; for example, the value 3 specifies the AMBER force field.

BMFF extends this mechanism as follows. If the first argument to the `FFLD` command is 10, an external process is started to generate force-field interactions. MacroModel obtains information it needs to initiate, control, and utilize the results of this process from an extended force-field file called `f`*nn*`.fld`, where *nn* is the value of this argument.

Values between 10 and 99, inclusive, are reserved for assignment by the MacroModel development team; for example, the value 10 is assigned to the MMFF force-field. Then the MMFF

force-field file would be called `f10.fld`. Customers developing their own in-house applications are free to use values of 100 or greater for *nn*.

The extended force-field file contains, on the first non-blank line, the word `Process:`, followed by a command line: typically, a process name plus (perhaps) command-line arguments. If the process name contains the character `/`, the named executable image is used; otherwise, the process with the given name is searched for first in the local directory and, if not found, in the `$SCHRODINGER/mmshare-v`*version*`/data` directory.

Following the `Process:` line, there may be one or more `Option:` lines. These specify command-line options for the server process that the user can invoke by means of arg4 of the `FFLD` command. Each `Option:` line should include an integer followed by one or more strings. If an integer given on an `Option:` line is specified in arg4 of `FFLD`, then the strings will be appended to the command-line when the process is started. Right now, only a single option specified in this manner can be used at run-time. Both the `Process:` and the `Option:` lines use free format; their contents consist of strings separated by spaces, but no tabs are allowed.

Beyond the first few lines, the extended force-field file follows the plan of a normal MacroModel force-field file. The `STR`, `BND`, etc. keywords define the equations which MacroModel is to use internally to describe stretches, bends, etc. MacroModel will call on the server, eventually, to obtain the atoms constituting the interactions and the corresponding parameters. For van der Waals and hydrogen-bonding interactions, the constituent atoms are passed to the server as needed, and the corresponding parameters are passed back whenever the pair-list is updated. Universal constants associated with the various interaction equations are specified on the equation lines for BMFF force-fields; see `f10.fld` for examples. In a BMFF force-field file,

The Main Interaction Section of the file may specify general interactions. If present, these will override the corresponding parameters obtained by the server. Similarly, the Substructure Section may contain overriding parameters for molecular substructures. The substructure section may also specify additional interactions (e.g., remote torsions) not created by the server process. If overriding parameters or added interactions are actually used, a warning message will be printed to the MacroModel log file. The server process may also write to the command file simply by writing to the UNIX standard output.

When the nonbonded pair list is updated, MacroModel (the client) passes each interacting atom-pair to the server and receives back the parameters for that pair; if overriding parameters exist for this pair, the server is not called. (If fast VDW processing is in effect, the client-side library stores a table of nonbonded parameters by pair type. In this situation, the server is never contacted for a nonbonded update. Here, this table is not consulted if overriding parameters exist for the pair in question.)

Before exiting, MacroModel commands the server to terminate.

E.3 MMFF Considerations

A BMFF server normally looks for its startup files (such as its standard parameter file) in the directory `$SCHRODINGER/mmshare-v`*version*`/data/f`*nn*, where *nn* is the value used in arg1 of the `FFLD` command to invoke the server. However, if the environment variable `MMFF_PATH` exists, `mmff_setup` expects to find a valid directory name stored in that variable, and looks for its run-time files there instead.

Format of .grd Files

Files with a `.grd` suffix are produced by the `DRIV` command. These files are used by Maestro to create contour plot displays.

Although right now Maestro and `DRIV` support two-dimensional data, the file format is designed to accommodate three dimensional data sets. The format is as follows. Fortran format descriptors are given for each part of the file. This file format is based on the Gaussian-90 electron-density file format.

Line 1: Format (A80). In general, this can contain any string, and can be used as a title line; however, if this line contains the string `BMIN` in the columns 1-4 followed by a list of atom numbers in (4I5) format, Maestro will attempt to display the molecule in file *filenameout*`.mae`, where *filename* is the prefix of the `.grd` file, and will use the atom numbers to associate the contour display with the torsion.

Line 2: Format (A80). As for line 1. The atom numbers define the second dihedral which was driven during the calculation.

Line 3: Format (I5,3F12.6). Natoms, X-Origin, Y-Origin, Z-Origin. Natoms is the number of atoms in the molecule. The X-Origin and Y-Origin variables define the axis values of the "origin" (bottom left) of the contour plot. These will be taken from the starting angles in the `DRIV` command; for example, they might both be –180.0.

Lines 4-6: Format (I5,3F12.6). Ninc, Xinc, Yinc, Zinc. Each of the three lines is for one driven angle. Ninc is the number of increments of the angle and Xinc, Yinc, and Zinc are the components of an increment along the axes of the first, second, and third driven angle. Since for `DRIV`-generated data the angles are driven separately and independently, and since we now support only the two-dimensional case, lines 4-6 should have the following form:

```
NN  NN.NNNNNNN    0.000000    0.000000
NN    0.000000  NN.NNNNNN    0.000000
 0    0.000000    0.000000    0.000000
```

`NN` is the number of increments of the angle and `NN.NNNNNN` is the size of an increment.

There next follow Natoms lines, starting with line 7. Each line has format (I5, 4F12.6), and the variables are atomic number, charge, and X, Y, and Z coordinates. These lines are actually ignored by MacroModel, but must be present.

Starting with line (7 + Natoms), the energetic data corresponding to the grid points are given, one energy value per line, in (F12.6) format. The energies are ordered so that the index corresponding to the first angle moves fastest, then the index corresponding to the second angle, then the third (if there is a third angle). So, for a two-dimensional data set with the energies in an array called E, the sequence of the energies would be E(1,1), E(2,1), ..., E(Ninc1,1), E(1,2), E(2,2), ..., E(Ninc1,2), ..., ..., E(1,Ninc2), E(2,Ninc2), ..., E(Ninc1,Ninc2). Thus, there are a total of Ninc1* Ninc2 energy values.

MINTA Background

G.1 Introduction

In this Appendix[2] we intend to put MINTA in perspective in the world of free-energy simulations. Until very recently, MD has been considered to be the only reasonable way of performing free-energy simulations (see for example Allen and Tildesley, 1987; McCammon and Harvey, 1987; Straatsma and McCammon, 1992). However, Jorgensen has shown that Metropolis Monte Carlo (MMC) simulation can be equally well utilized for proteins and pointed out that "Monte Carlo sampling of the internal coordinates of protein side chains is likely to be as efficient as molecular dynamics" (Essex et al., 1997). Of course, one has to be very careful with the term Monte Carlo simulation. MMC sampling in the context of free-energy perturbation is fundamentally different from MC conformational searching/sampling. The conformational search aspect of MC sampling for protein side chains (Shenkin et al., 1996) and for enzyme active sites (Keseru and Kolossváry, 1997) was in fact recognized earlier. Of course, conformational analysis itself does not provide "free energy," but MINTA does.

Moreover, it was pointed out by Guida and coworkers (1992) that "presently, thermodynamic perturbation and integration methods seem to be limited to situations in which the structural perturbation considered does not induce a significant conformational change," and it was also noted by Kollman (1996) that "the largest challenge facing us is the local minimum, or sampling, problem" and "it is clear that a standard application of molecular dynamics and Monte Carlo methods is very inefficient at traversing the space between local minima." Free-energy simulations using MD or MMC have been applied most successfully to macromolecular complexes where the X-ray structure of the bound complex is known. Although the sampling during MD or MMC simulations allows for the free movement of the ligand, because of the high energetic barriers that impede ligand movement in the active site, MD and MMC tend to keep the ligand close to its bound X-ray conformation, and only the ligand's internal degrees of freedom are sampled adequately. This approach has been proven, nonetheless, very successful in calculating the binding free-energy difference between similar ligands in a relatively rigid active site. However, in a common situation where the bound conformation is not known or the ligand can adopt multiple binding conformations, or when the protein host undergoes significant backbone (e.g., loop) conformational changes upon binding, MD and MMC

2. Excerpted with permission from Keseru, Gy.; Kolossváry, I. *Molecular Mechanics and Conformational Analysis in Drug Design;* Blackwell Science: Oxford, 1999; Appendix to Chapter 7. Ordering information: http://www.blackwell-science.com/~cgilib/bookpage.bin?File=5798

alone cannot presently provide converged free-energy simulation results. Conformational analysis is therefore a prerequisite for any free-energy calculation involving highly flexible docking (especially if three-dimensional structural data are not available), whether it is based on MD or MMC simulation, or a direct approach such as MINTA.

Before starting a detailed discussion of free-energy simulation techniques and the place for MINTA among them, let us provide recent literature references to the state-of-the-art of calculating protein-ligand binding affinities. Reviews include Kollman (1993), Ajay and Murcko (1995), Gilson et al. (1997), Lamb and Jorgensen (1997), and Babine and Bender (1997).

G.2 The Thermodynamic Cycle

The thermodynamic cycle is based on the fact that free energy is a state function. For example we want to calculate the binding free-energy difference between two ligands with respect to the same macromolecular host in solution. The corresponding thermodynamic cycle can be written as follows:

$$\begin{array}{ccc} \mathrm{H(s)} + \mathrm{L_A(s)} & \xrightarrow{\Delta G_1} & \mathrm{HL_A(s)} \\ \Delta G_3 \downarrow & & \downarrow \Delta G_4 \\ \mathrm{H(s)} + \mathrm{L_B(s)} & \xrightarrow{\Delta G_2} & \mathrm{HL_B(s)} \end{array} \qquad \Delta\Delta G_{\mathrm{AB}} = \Delta G_1 - \Delta G_2 = \Delta G_3 - \Delta G_4 \tag{1}$$

The natural, chemical path would be along the horizontal arrows (ΔG_1-ΔG_2) corresponding to the actual chemical complex formation. However, the associated simulations are unfortunately subject to excessive statistical error rendering this approach extremely unstable and virtually useless. On the other hand, the counterintuitive path along the vertical arrows (ΔG_3-ΔG_4) is non-chemical, nonetheless, it is well suited for simulations. The computational procedure often referred to as computational alchemy involves “mutating” L_A into L_B once in pure solvent and then again in the binding cavity of the protein host H. The numerical values of ΔG_3 and ΔG_4 can be obtained by many different ways based on different levels of approximation, which will be discussed below.

Equation (1) is used to calculate relative binding free energies. The thermodynamic cycle, however, can be used to estimate absolute free energies as well. This procedure is termed double annihilation and the corresponding thermodynamic cycle can be written as a “balance” equation, as in Equation (2).

$$\begin{array}{lll} & \mathrm{L(s)} \rightarrow \varnothing\mathrm{(s)} & \Delta G_{\mathrm{L(s)} \rightarrow \varnothing\mathrm{(s)}} \\ \mathrm{H(s)} & \rightarrow \mathrm{HL(s)} & -\Delta G_{\mathrm{HL(s)} \rightarrow \mathrm{H(s)}} \\ \hline \mathrm{H(s)} + \mathrm{L(s)} & \rightarrow \mathrm{HL(s)} & \Delta G_{\mathrm{b}}^{\mathrm{abs}} \end{array} \tag{2}$$

Equation (2) describes the balance of double annihilation, i.e., in one process the ligand vanishes from the solvent, and in the other process the same ligand vanishes from the ligand-host complex. The net ΔG_{b} is the absolute binding free energy of ligand L to the host H. Note, however that the vanishing of the ligand is intrinsically coupled with the loss of translational and rotational freedom. Hermans and Wang (1997) have recently introduced a restoring potential applied to the ligand that allows for the correct inclusion of the loss of translational and rotational freedom in double annihilation binding free-energy calculations.

G.3 Computational Methods

The computational methods include a wide palette of approximations. On one end, there are the popular docking algorithms based on empirical scoring functions (to estimate binding free energies) that allow the fast screening of thousands of ligands in the active site of an enzyme model on the time scale of only CPU minutes. Such methods include Glide, DOCK (DesJarlais et al., 1988), AutoDock (Goodsell and Olson, 1990; Morris et al., 1996), LUDI (Böhm, 1994), QXP or McDock (McMartin and Bohacek, 1995; McMartin and Bohacek, 1997), Hammerhead (Welch et al., 1996), and SMoG (DeWitte and Shakhnovich, 1996). The primary use of these methods is in the lead finding phase of drug design. Lead finding involves screening of large databases (tens or even hundreds of thousands of small molecules) for potential drug candidates, which hopefully bind to a particular enzyme. The fast screening procedure docks the ligands into the enzyme active site model, and the scoring is based on simple empirical functions that can reproduce experimental relative binding affinities within 1 to 2 kcal/mol.

On the “high-end” of the list there are highly sophisticated methods based on statistical perturbation theory, in particular free-energy perturbation (FEP) and thermodynamic integration (TI). These methods are slow, but represent the highest level of accuracy—~0.5 kcal/mol for relative binding affinities—that can be achieved today and, therefore, are primarily used for lead optimization. Lead optimization in essence means that small changes are applied to the lead molecules obtained during the lead finding process, trying to enhance their binding affinity by one to three orders of magnitude.

FEP establishes a link between free-energy difference ΔG and potential energy difference ΔE utilizing Zwanzig's famous equation (Zwanzig, 1954):

$$\Delta G_{AB} = -RT \ln \left\langle \exp\left(-\frac{E_B - E_A}{RT}\right)\right\rangle_A \tag{3}$$

ΔG_{AB} is the free-energy difference between two systems, "A" and "B," which can, e.g., represent two different ligands in a situation described by the thermodynamic cycle in Equation (1). E_A and E_B are the potential energy (molecular mechanics energy including solvation energy) of system "A" and "B," respectively. Note that E_A and E_B are functions of the coordinates of the two different systems. The bracket $\langle\,\rangle_A$ refers to an ensemble average over system "A," which is determined at a particular temperature T (R is the gas constant).

It should be noted that Equation (3) is exact. It will be discussed later why is FEP a perturbation theory. For the sake of argument, let us assume that we want to use Equation (3) directly to calculate the binding free-energy difference between two ligands L_A and L_B with respect to a macromolecular host H. What does it mean to compute an ensemble average over system "A"? The answer depends on what kind of simulation is applied. If it is MD, the answer is that the MD simulation is running using L_A, L_A is replaced by L_B after every time step, and the exponential in Equation (3) is accumulated during the full course of the MD simulation. MD (stochastic dynamics, to be correct) mimics thermal motion in a thermal bath and, therefore, generates the so-called canonical ensemble, i.e., samples the configuration space with the Boltzmann probability. Thus, the simple arithmetic mean of the accumulated exponentials $\exp(-(E_B-E_A)/RT)$ gives the ensemble average in Equation (3).

In case of MMC simulation, the Metropolis algorithm guaranties that the canonical ensemble is generated (Metropolis et al., 1953). With MMC, therefore, the well-known Metropolis criterion is applied to L_A to drive the simulation. Similar to MD, the ensemble average is calculated as the arithmetic mean of the accumulated exponentials $\exp(-(E_B-E_A)/RT)$ where L_A is temporarily replaced by L_B after each accepted MMC move. It is important to note that FEP is always carried out in an explicit solvent box.

Thermodynamic integration provides an alternative way for free-energy perturbation and is based on the following formula (Kollman, 1993):

$$\Delta G_{AB} = \int_{\rho = A}^{\rho = B} \left\langle \frac{\partial E_\rho}{\partial \rho} \right\rangle_\rho d\rho \tag{4}$$

where the ensemble average $\langle\,\rangle_\rho$ of the derivative of the energy with respect to ρ is evaluated at various values of ρ, and the outer integral is solved numerically. ρ represents a "reaction coordinate," which is in most cases non-physical, but delineates a computationally accessible

path in phase space to perturb system "A" into system "B." The ensemble average itself can be calculated the same way as described with FEP.

It is now time to shed light on the perturbation nature of FEP. Although Equation (3) is exact, evaluation of the ensemble average for systems which differ in more than a trivial way, must be carried out in numerous intermediate stages. The problem is that the "replacement" step in the simulation grossly brakes the thermodynamic equilibrium if, e.g., L_B is significantly different from L_A. Consequently, Equation (3) will not lead to a sensible free energy. The solution to this problem is making FEP a perturbation procedure by breaking the "mutation" of L_A into L_B in several small steps. In other words, one breaks up the free-energy calculation into windows, each one involving only a small change whose free-energy contribution can be calculated accurately using Equation (3). Since free energy is a state function, one can add up the small contributions of the subsequent windows to obtain an accurate estimate of the overall ΔG:

$$\Delta G_{AB} = \sum_{\lambda = A}^{\lambda = B} -RT \ln \left\langle \exp\left(-\frac{\Delta E_\lambda}{RT}\right)\right\rangle_\lambda \tag{5}$$

where λ is a parameter, which is used to mutate system "A" into system "B" smoothly. ΔE_λ is the potential energy difference between two slightly different, intermediate forms of the hybrid molecule "AB." Mutation simply means that the geometry and the force field parameters of "A" are continuously changed in small subsequent steps into that of "B." For example, a -CH_2-OH fragment can be changed to a -CH_2-SH fragment by continuously interpolating the force field parameters between O and S, and adjusting the geometry to adopt longer bond lengths for SH. One can also change the chirality of an atom by continuously interchanging two substituents. The mutation process often requires dummy atoms that vanish in one molecule, but slowly emerge as real atoms in the other molecule.

In summary, the thermodynamic cycle in Equation (1) requires two series of FEP simulations following the computational alchemy path. One series will mutate L_A into L_B in solvent and the other series will carry out the same mutation in the cavity of the protein. Note that both series consist of several MD or MMC simulations, each of which should sample Boltzmann-weighted ensembles of conformational states adequately to yield converged and accurate free-energy values. Therefore, FEP is an extremely time consuming procedure which is always suspect of inadequate sampling (van Gunsteren and Mark, 1992).

A very promising approach termed the linear response method (LRM) (Åqvist et al., 1994; Åqvist, 1996) eliminates the substantial effort devoted to FEP simulations at intermediate points ($0<\lambda<1$) along the mutation path. LRM only requires simulations at the end points with the pure systems "A" and "B." LRM is a semi-empirical method, which is based on the quadratic dependence of free energy on solute charge in the Born model for ion solvation. It can be shown that in this model the electrostatic contribution toward the solvation energy is

equal to half of the corresponding ion-solvent interaction energy (Warshel and Russell, 1984; Roux et al., 1990). LRM extends this idea toward protein-ligand systems by linking free-energy changes to the interaction energies between solutes (ligands) and their environment (protein). The interactions are broken down into electrostatic and van der Waals contributions. The free-energy difference between systems "A" and "B" is given by:

$$\Delta G_{AB} = \frac{1}{2}\left(\langle E_{ele}\rangle_A - \langle E_{ele}\rangle_B\right) + \alpha\left(\langle E_{vdw}\rangle_A - \langle E_{vdw}\rangle_B\right) \tag{6}$$

where α is an empirical parameter derived from experimental binding data and the ensemble averages $\langle E_{ele}\rangle$ and $\langle E_{vdw}\rangle$ are obtained via short MD simulations in a periodic box of water on system "A" and "B," respectively. LRM has been recently applied successfully to various substrates of cytochrome $P450_{cam}$ (Paulsen and Ornstein, 1996) and HIV-1 protease inhibitors (Hultén et al., 1997).

An extended linear response method has been recently introduced (Carlson and Jorgensen, 1995; McDonald et al., 1997) and applied to predicting binding affinities for sulfonamide inhibitors with human thrombin (Jones-Hertzog and Jorgensen, 1997). The extended LRM includes a cavity term and also allows the factor of 0.5 for electrostatic interactions to vary. The resulting free-energy equation is

$$\begin{aligned}\Delta G_{AB} = {} & \beta\left(\langle E_{ele}\rangle_A - \langle E_{ele}\rangle_B\right) + \alpha\left(\langle E_{vdw}\rangle_A - \langle E_{vdw}\rangle_B\right) \\ & + \gamma\left(\langle \mathrm{SASA}\rangle_A - \langle \mathrm{SASA}\rangle_B\right)\end{aligned} \tag{7}$$

where all three empirical parameters (α, β, γ) are derived from experimental binding data and the third term includes a contribution from the solute's (ligand) solvent-accessible surface area (SASA).

A novel smart Monte Carlo approach termed jumping-between-wells (JBW) should also be mentioned (Senderowitz et al., 1995; Senderowitz et al., 1997). The JBW method coupled with molecular dynamics involves directly monitoring the populations of various conformations of the two systems "A" and "B" in a simulation in which conformational interconversions occur frequently, producing converged, Boltzmann-weighted ensembles of conformational states. JBW uses a "lookup table" of low-energy conformational states obtained by a preceding conformational search to direct the simulation toward low-energy regions of the PES. Therefore, JBW is not impeded by high barriers like MD, it can simply jump from one minimum to another (subject to the Metropolis criterion) and sample the energy wells locally via MD and MMC. The free-energy difference between two systems can be calculated directly from monitoring the population ratio of the different conformational states of "A" and "B" during the simulation. Note that JBW does not require mutations either, since the simulation is carried out directly on the pure systems.

Direct methods have emerged very recently, which involve the direct evaluation of the configuration integral as sums over conformational minima (Equation (1)). Most notably, a new method termed "mining minima" has been introduced in which the configuration integral is evaluated over the "soft modes" identified as torsion angles (Head et al., 1997). It should be stressed, however, that the exclusion of "hard modes," such as bond lengths and bond angles, is in general a poor approximation. Cyclic structures, e.g., undergo sufficient variation of their ring bond angles and even bond lengths during conformational interconversions, to contribute a significant amount to the conformational free energy. Therefore, a direct method such as MINTA is sought that can evaluate the configuration integral in all degrees of freedom, in order to calculate accurate free energies.

On this palette MINTA can be placed between LRM and FEP. Evaluation of the thermodynamic cycle in Equation (1) requires four MINTA calculations on L_A, L_B, HL_A, and HL_B, respectively. One MINTA calculation including the conformational search is comparable to a converged MD or MMC simulation on the same system. Therefore, MINTA is intrinsically faster than FEP. On the other hand, MINTA is slower than LRM, because the latter does not require a fully converged simulation. It is too early to make a fair comparison of the accuracy of MINTA vs. FEP or LRM, but based on our results so far, we expect MINTA to be between LRM and FEP in this respect, too. MINTA is certainly subject to two liabilities: (i) the focus on low-energy conformations and (ii) the use of a continuum solvation model. However, the high barriers in a protein-ligand complex with respect to ligand movement in the cavity clearly provide justification for (i) (advantage rather than liability), and continuum solvation models such as the GBSA model (Still et al., 1990; Qiu et al., 1997) have recently undergone significant improvements (including the continuum treatment of long-range interactions with application to protein-ligand binding (Simonson et al., 1997)) to render them on par with explicit models for many systems, thus alleviating (ii). In summary, we believe that MINTA should find wide utility as a simple tool for medicinal chemists already familiar with conformational analysis.

G.4 References

Ajay, Murcko, M. A. *J. Med. Chem.* **1995,** *38*, 4953.

Allen, M. P.; Tildesley, D. J. *Computer Simulation of Liquids;* Clarendon Press: Oxford, 1987.

Åqvist, J. *J. Comput. Chem.* **1996,** *17*, 1587.

Åqvist, J.; Medina, C.; Samuelsson, J.-E. *Protein Engng.* **1994,** *7*, 385.

Babine, R. E.; Bender, S. L. *Chem. Rev.* **1997,** *97*, 1359.

Böhm, H.-J. *J. Comput.-Aided Mol. Design* **1994,** *8*, 243.

Carlson, H. A.; Jorgensen, W. L. *J. Phys. Chem.* **1995,** *99*, 10667.

DesJarlais, R. L.; Sheridan, R. P.; Seibel, G. L.; Dixon, J. S.; Kuntz, I. D.; Venkataraghavan, R. *J. Med. Chem.* **1988,** *31*, 722.

DeWitte, R. S.; Shakhnovich, E. I. *J. Am. Chem. Soc.* **1996,** *118*, 11733.

Essex, J. W.; Severance, D. L.; Jorgensen, W. L. *J. Phys. Chem. B* **1997,** *101*, 9663.

Gilson, M. K.; Given, J. A.; Bush, B. L.; McCammon, J. A. *Biophys.* **1997,** J. *72*, 1047.

Goodsell, D. S.; Olson, A. J. *Proteins* **1990,** *8*, 195.

Guida, W. C.; Bohacek, R. S.; Erion, M. D. *J. Comput. Chem.* **1992,** *13*, 214.

Head, M. S.; Given, J. A.; Gilson, M. K. *J. Phys. Chem. A* **1997,** *101*, 1609.

Hermans, J.; Wang, L. *J. Am. Chem. Soc.* **1997,** *119*, 2707.

Hultén, J.; Bonham, N. M.; Nillroth, U.; Hansson, T.; Zuccarello, G.; Bouzide, A.; Åqvist, J.; Classon, B.; Danielson, U. H.; Karlén, A.; Kvarnström, I.; Samuelsson, B.; Hallberg, A. *J. Med. Chem.* **1997,** *40*, 885.

Jones-Hertzog, D. K.; Jorgensen, W. L. *J. Med. Chem.* **1997,** *40*, 1539.

Jorgensen, W. L. *Acc. Chem. Res.* **1989,** *22*, 184.

Keser, G. M.; Kolossváry, I.; Bertók, B. *J. Am. Chem. Soc.* **1997,** *119*, 5126.

Kollman, P. A. *Chem. Rev.* **1993,** *93*, 2395.

Kollman, P. A. *Acc. Chem. Res.* **1996,** *29*, 461.

Lamb, M. L.; Jorgensen, W. L. *Curr. Op. Chem. Biol.* **1997,** *1*, 449.

McCammon, J. A.; Harvey, S. C. *Dynamics of Proteins and Nucleic Acids;* Cambridge University Press; Cambridge, 1987.

McDonald, N. A.; Carlson, H. A.; Jorgensen W. L. *J. Phys. Org. Chem.* **1997,** *10*, 563.

McMartin, C.; Bohacek, R. S. *J. Comput.-Aided Mol. Design* **1995,** *9*, 237.

McMartin, C.; Bohacek, R. S. *J. Comput.-Aided Mol. Design* **1997,** *11*, 333.

Metropolis, N.; Rosenbluth, A. W.; Rosenbluth, M. N.; Teller, A.; Teller E. *J. Chem. Phys.* **1953,** *21*, 1087.

Morris, G. M.; Goodsell, D. S.; Huey, R.; Olson, A. J. *J. Comput.-Aided Mol. Design* **1996,** *10*, 293.

Paulsen, M. D.; Ornstein, R. L. *Protein Engng.* **1996,** *9*, 567.

Qiu, D.; Shenkin, P. S.; Hollinger, F. P.; Still, W. C. *J. Phys. Chem. A* **1997,** *101*, 3005.

Roux, B.; Yu, H.-A.; Karplus, M. *J. Phys. Chem.* **1990,** *94*, 4683.

Senderowitz, H.; Guarnieri, F.; Still, W. C. *J. Am. Chem. Soc.* **1995,** *117*, 8211.

Senderowitz, H.; McDonald, D. Q.; Still, W. C. *J. Org. Chem.* **1997,** *62*, 9123.

Shenkin, P. S.; Farid, H.; Fetrow J. S. *Proteins: Struct., Funct., Genet.* **1996,** *26*, 323.

Simonson, T.; Archontis, G.; Karplus, M. *J. Phys. Chem. B* **1997,** *101*, 8349.

Still, W. C.; Tempczyk, A.; Hawley, R. C.; Hendrickson, T. *J. Am. Chem. Soc.* **1990,** *112*, 6127.

Straatsma, T. P.; McCammon, J. A. *Ann. Rev. Phys. Chem.* **1992,** *43*, 407.

Van Gunsteren, W. F.; Mark, A. E. *Eur. J. Biochem.* **1992,** *204*, 947.

Warshel, A.; Russell, S. T. *Q. Rev. Biophys.* **1984,** *17*, 283.

Welch, W.; Ruppert, J.; Ajay, N. J. *Chem. Biol.* **1996,** *3*, 449.

Zwanzig, R. W. *J. Chem. Phys.* **1954,** *22*, 1420.

References

1. Allinger, N. L. Conformational Analysis 130. MM2. A Hydrocarbon Force Field Utilizing V1 and V2 Torsional Terms. *J. Am. Chem. Soc.* **1977,** *99*, 8127.

2. Allinger, N. L.; Yuh, Y. H. Quantum Chemistry Program Exchange, Bloomington, Indiana, Program No. 395, Molecular Mechanics. *Molecular Mechanics;* Burkert, U., Allinger, N. L., eds.; ACS Monograph 177; American Chemical Society: Washington, DC, 1982.

3. Allinger, N. L.; Yuh, Y. H.; Lii, J-H Molecular Mechanics. The MM3 Force Field for Hydrocarbons. 1. *J. Am. Chem. Soc.* **1989,** *111*, 8551.

4. Weiner, S. J.; Kollman, P.; Case, D.; Singh, U.; Ghio, C.; Alagona, G.; Profeta, S.; Weiner, P. A new Force Field for Molecular Mechanical Simulation of Nucleic Acids and Proteins. *J. Am. Chem. Soc.* **1984,** *106*, 765.

5. Weiner, S. J.; Kollman, P. A.; Nguyen, D. T.; Case, D. A. An all atom force field for simulations of proteins and nucleic acids. *J. Comput. Chem.* **1986,** *7*, 230.

6. Cornell, W. D.; Cieplak, P.; Bayly, C. I.; Gould, I. R.; Merz, K. M.; Ferguson, D. M.; Spellmeyer, D. C.; Fox, T.; Caldwell, J. W.; Kollman, P. A. A second generation force field for the simulation of proteins, nucleic acids and organic molecules. *J. Am. Chem. Soc.* **1995,** *117*, 5179.

7. Jorgensen, W. L.; Tirado-Rives, J. The OPLS Potential Functions for Proteins. Energy Minimization for Crystals of Cyclic Peptides and Crambin. *J. Am. Chem. Soc.* **1988,** *110*, 1657.

8. Halgren, T. A. Merck Molecular Force Field. I. Basis, Form, Scope, Parameterization and Performance of MMFF94. *J. Comput. Chem.* **1996,** *17,* 490-519.

9. Halgren, T. A. Merck Molecular Force Field. II. MMFF94 van der Waals and Electrostatic Parameters for Intermolecular Interactions. *J. Comput. Chem.* **1996,** *17,* 520-552.

10. Halgren, T. A. Merck Molecular Force Field. III. Molecular Geometrics and Vibrational Frequencies for MMFF94. *J. Comput. Chem.* **1996,** *17,* 553-586.

11. Halgren, T. A.; Nachbar, R. B. Merck Molecular Force Field. IV. Conformational Energies and Geometries. *J. Comput. Chem.* **1996,** *17,* 587-615.

12. Halgren, T. A. Merck Molecular Force Field. V. Extension of MMFF94 using Experimental Data, Additional Computational Data and Empirical Rules. *J. Comput. Chem.* **1996,** *17,* 616-641.

13. Halgren, T. A. MMFF VI. MMFF94s Option for Energy Minimization Studies. *J. Comput. Chem.* **1999,** *20,* 720-729.

14. Halgren, T. A. MMFF VII. Characterization of MMFF94, MMFF94s and Other Widely Available Force Fields for Conformational Energies and for Intermolecular Interaction Energies and Geometries. *J. Comput. Chem.* **1999,** *20,* 730-748.

15. Jorgensen, W. L.; Maxwell, D. S.; Tirado-Rives, J. Development and Testing of the OPLS All-Atom Force Field on Conformational Energetics and Properties of Organic Liquids. *J. Am Chem. Soc*. **1996,** *118*, 11225-11236.

16. Hasel, W.; Hendrickson, T. F.; Still, W. C. A Rapid Approximation to the Solvent Accessible Surface Areas of Atoms. *Tetrahedron Comput. Methodol.* **1988,** *1*, 103.

17. Ooi, T.; Oobatake, M.; Nemethy, G.; Scheraga, H. A. Accessible Surface Areas as a Measure of the Thermodynamic Parameters of Hydration of Peptides. *PNAS* **1987,** *84,* 3086.

18. Still, W. C.; Tempczyk, A.; Hawley, R. C.; Hendrickson, T. Semianalytical treatment of solvation for molecular mechanics and dynamics. *J. Am. Chem. Soc.* **1990,** *112*, 6127.

19. *Cheng, A.;* Best, S. A.; Merz, K. M. Jr.; Reynolds, C. H. GB/SA water model for the Merck molecular force field (MMFF). *J. Mol. Graphics Modell.* **2000**, *18*, 273–282.

20. Best, S. A.; Merz, K. M., Jr.; Reynolds, C. H. A GB/SA Based Continuum Solvation Model for Octanol. *J. Phys. Chem. B* **1997***, 101*, 10479-10487.

21. Hay, B. P. Methods for molecular mechanics modeling of coordination compounds. *Coord. Chem. Rev.* **1993,** *126*, 177-236.

22. Polak, E.; Ribiere, G. Note sur la Convergence de Methodes de Directions Conjuguees. *Revue Francaise Inf. Rech. Oper., Serie Rouge* **1969,** *16-R1*, 35.

23. Oren, S. S.; Spedicato, E. Optimal conditioning of self-scaling variable metric algorithms. Mathematical Programming. *Math. Programming* **1976,** *10*, 70.

24. Ponder, J. W.; Richards, F. M. An Efficient Newton-like Method for Molecular Mechanics Energy Minimization of Large Molecules. *J. Comput. Chem.* **1987,** *8*, 1016.

25. Culot, P.; Dive, G.; Nguyen, V. H.; Ghuysen, J. M. A Quasi-Newton Algorithm for FirstOrder Saddle Point Location. *Theor. Chim. Acta* **1992,** *82*, 189.

26. Sefler, A. M.; Lauri, G.; Bartlett, P. A. A Convenient Method for Determining Cyclic Peptide Conformation from 1D 1H-NMR Information. *Int. J. Pept. Protein Res.* **1996,** *48*, 129.

27. Kolossváry, I.; Guida, W. C. Low Mode Search. An efficient, automated computational method for conformational analysis: application to cyclic and acyclic alkanes and cyclic peptides. *J. Am. Chem. Soc.* **1996,** *118*, 5011.

28. Kolossváry, I.; Guida, W. C. Low-mode Conformational Search Elucidated. Application to C39H80 and Flexible Docking of 9-Deazaguanine Inhibitors to PNP. *J. Comp. Chem.* **1999,** *20*, 1671.

29. Kolossváry, I.; Keseru, G. M. Hessian-Free Low-Mode Conformational Search for Large-Scale Protein Loop Optimization: Application to c-jun N-Terminal Kinase JNK3. *J. Comput. Chem.* **2001,** *22*, 21.

30. Keseru, G. M.; Kolossváry, I. Fully Flexible Low-Mode Docking: Application to Induced Fit in HIV Integrase. *J. Am. Chem. Soc.* **2001,** *123*, 12708.

31. Shenkin, P. S.; Yarmush, D. L.; Fine, R. M.; Wang, H.; Levinthal, C. Predicting antibody hypervariable loop conformation. I. Ensembles of random conformations for ring-like structures. *Biopolymers* **1987**, *26*, 2053-2085.

32. Fine, R. M.; Wang, H.; Shenkin, P. S.; Yarmush, D. L.; Levinthal, C. Predicting Antibody Hypervariable Loop Conformations. II: Minimization And Molecular Dynamics Studies Of MCPC603 From Many Randomly Generated Loop Conformations. *Proteins* **1986**, *1*, 342.

33. Sorensen, D. C. *ARPACK Tutorial: Implicitly Restarted Arnoldi/Lanczos Methods for Large Scale Eigenvalue Calculations;* Rice University: Houston, TX, 1995.

34. Lehoucq, R. B.; Sorensen, D. C.; Yang, C. *ARPACK User's Guide: Solution of Large Scale Eigenvalue Problems With Implicitly Restarted Arnoldi Methods;* Rice University: Houston, TX, 1997.

35. Chang, G.; Guida, W. C.; Still, W. C. An internal coordinate Monte-Carlo method for searching conformational space. *J. Am. Chem. Soc.* **1989,** *111*, 4379.

36. Saunders, M.; Houk, K. N.; Wu, Y.-D.; Still, C. W.; Lipton, M.; Chang, G.; Guida, W. C. Conformations of Cycloheptadecane: A Comparison of Methods for Conformational Searching. *J. Am. Chem. Soc.* **1990,** *112*, 1419.

37. Goodman, J. M.; Still, W. C. Searching Conformation Space. *J. Comput. Chem.* **1991,** *12*, 1110.

38. Li, Z.; Scheraga. H. Monte Carlo-minimization approach to the multiple-minima problem in protein folding. *PNAS* **1987,** *84*, 6611.

39. Ryckaert. J.-P. Special geometrical constraints in the molecular dynamics of chain molecules. *Mol. Phys.* **1985,** *55,* 549.

40. Berendsen, H. J. C.; Postma, J. P. M.; van Gunsteren, W. F.; DiNola, A.; Haak, J. R. Molecular dynamics with coupling to an external bath. *J. Chem. Phys.* **1984,** *81*, 3684.

41. van Gunsteren, W. F.; Berendsen H. J. C. A leap-frog algorithm for stochastic dynamics. *Mol. Simul.* **1988,** *1*, 173.

42. Kolossváry, I. Evaluation of the Molecular Configuration Integral in All Degrees of Freedom for the Direct Calculation of Conformational Free Energies: Prediction of the Anomeric Free Energy of Monosaccharides. *J. Phys. Chem. A* **1997,** *101*, No. 51, 9900-9905.

43. Kolossváry, I. Evaluation of the Molecular Configuration Integral in All Degrees of Freedom for the Direct calculation of Binding Free Energies: Application to the Enantioselective Binding of Amino Acid Derivatives to Synthetic Host Molecules. *J. Am. Chem. Soc.* **1997,** *119*, 10233-10234.

44. Kolossváry, I. Mode Integration (MINTA): A Method for Selecting a Molecule Based on Conformational Free Energy of One Molecule for Another Molecule. U.S. Patent 08/940, 145, September 29, 1997.

45. Keseru, Gy.; Kolossváry, I. *Molecular Mechanics and Conformational Analysis in Drug Design;* Blackwell Science: Oxford, 1999; Chapter 7.

46. Williams, D. H. Structural studies on some antibiotics of the vancomycin group, and on the antibiotic-receptor complexes, by proton NMR. *Acc. Chem. Res.* **1984,** *17*, 364.

47. Cohen, M. L. Epidemiology of drug resistance: Implications for a post- antimicrobial era. *Science (Washington, D.C.)* **1992,** *257*, 1050.

48. Mackay, J. P.; Gerhard, U.; Beauregard, D. A.; Westwell, M. S.; Searle, M. S.; Williams, D. H. Glycopeptide Antibiotic Activity and the Possible Role of Dimerization: A Model for Biological Signaling. *J. Am. Chem. Soc.* **1994,** *116*, 4581.

49. Dunayevskiy, Y. M.; Lyubarskaya, Y. V.; Chou, Y.-H.; Vouros, P.; Karger, B. L. Simultaneous Measurement of Nineteen Binding Constants of Peptides to Vancomycin Using Affinity Capillary Electrophoresis-Mass Spectrometry. *J. Med. Chem.* **1998,** *41*, 1201.

50. Slater,J.C.; Kirkwood, J.G. The van der Waals forces in gases. *Phys. Rev.* **1931,** *37*, 682.

51. Weiner, P. K.; Kollman, P. A. AMBER: Assisted Model Building with Energy Refinement. A General Program for Modeling Molecules and Their Interactions. *J. Comput. Chem.* **1981,** *2*, 287.

52. Kaminski, G. A.; Friesner, R. A.; Tirado-Rives, J.; Jorgensen, W. J. *J. Phys. Chem. B* **2001**, 105, 6474.

53. Weiser, J.; Holthausen, M. C.; Fitjer, L. *J. Comp. Chem.* **1997**, *18*, 1264.

Opcode Index

Bold face page numbers indicate the opcode definition. Duplicate page numbers indicate that the opcode is referred to in more than one opcode definition section.

A

ADDC 5, **75**
ALGN 6, 81, 82, **82**, 139
  DEBG GV 146
  with MCOP arg7 105
ARPK 53, 91, **98**, 149
ASET 3, **48**, 50, 139
ASL1 ..... 31, 65, **66**
ASL2 ..... 31, 65, **66**
ASL3 ..... 31, 65, **67**
ASNT 3, **50**, 51
ATEQ 74, **78**, 144, 172
AUOP 29, **87**, 158
AUTO .. 5, 41, 52, 76, **84**, 87, 88, 91, 101, 103, 152
  DEBG GV 147
  with LMC2 91
  with LMCS 88
  with MCMM 101
  with MINI 52

B

BDCO 3, **38**, 46, 145, 195
BGIN . 8, 32, 41, **44**, 44, 45, 49, 50, 52, 63, 82, 83, 101, 103, 108, 135, 136, 169

C

CGEN . 5, 8, 20, 21, 23, 28, 87, 146, 155, **155**, 157, 159, 161, 162
CGO2 ..... 87, **157**, 163
CGO3 ..... **159**, 163
CGO4 ..... **160**, 163
CGO5 ..... **162**, 163
CGO6 ..... **163**
CGOP **155**, 160, 162, 163
CHGF 28, **42**, 42, 145, 172
CHIG ... 52, **76**, 85, 88, 95, 96, 101, 104, 108, 111, 146
CHYD **164**
COMP 52, 74, 75, **76**, 78, 79, 80, 83, 88, 95, 96, 133
CONV 28, **55**
COPY 6, 81, **82**, 82, 139
  DEBG GV 146
CRMS 76, **77**

D

DEBG 1, 4, 8, 15, 17, 18, 32, 39, 42, 45, 46, 48, 51, 55, 56, 57, 62, 63, 64, 67, 69, 72, 74, 78, 79, 83, 86, 88, 96, 98, 111, 118, 122, 123, 124, 127, 128, 138, 140, **141**, 150, 152, 172, 174, 194, 214
  45 (new) 143
  55 (new) 143
DEMX 29, 54, 75, **80**, 88, 101, 106, 113
DISC **115**
DLST **47**
DRIV **63**, 142, 225
DUMP **149**

E

ELST 16, **45**, 48, 49, 56, 60, 63, 139, 145
END 32, 41, 44, **44**, 45, 49, 50, 52, 63, 82, 83, 101, 103, 108, 135, 136, 169
EXN2 ..... 2, **38**, 46
EXNB .. 2, 28, **37**, 38, 39, 40, 46, 53, 113, 129, 172

F

FEAV 7, 21, **135**, 152
FEGA 7, **135**, 136
FEIA 7, **135**
FESA 7, 21, 135, **136**, 152
FESU 7, **137**
FFLD 4, 16, 28, **33**, 36, 75, 169, 172, 189, 221, 222, 223
FFOP **36**
FLAP 6, **106**, 107
FXAT ... 3, 4, 64, 65, 66, 67, **67**, 95, 111, 112, 135, 141, 151, 172, 194
FXBA 3, 50, **70**, 95, 141, 172
FXCO **72**
FXDI 3, 50, **69**, 95, 113, 135, 141, 172
FXHB **73**
FXTA 3, 50, **71**, 72, 95, 116, 141, 172

G

GEOM 9, **149**

H

HADD **45**, 45
HDEL **45**

I

IMCC **133**
IMPO **131**
IMPS 109, 114, 128, **130**, 131, 132, 133, 137, 145, 146
ITBS **133**
ITOR (obsolete) **131**

J

JRED 150, **151**
JWRT **150**, 152

L

LIGB 5, 51, 101, 113, **114**, 141
LMC2 . 3, 5, 8, 9, 20, 21, 23, 28, 44, 76, 78, 84, **90**, 98, 99, 103, 105, 144, 148, 152, 153
LMCS . 3, 5, 8, 9, 20, 21, 23, 28, 44, 58, 59, 61, 76, 78, 83, 84, 86, **87**, 88, 90, 91, 92, 93, 103, 104, 105, 106, 139, 144, 148, 152, 153
LOGP 7, **41**
    DEBG GV 147
LOOP 5, 9, 76, 84, **93**, 96, 103, 105
    DEBG GV 147
LPOP **96**

M

MBAE 5, 6, 8, 20, 21, 23, 44, 46, 49, 65, 68, 70, 71, 82, 84, **138**, 152
    DEBG GV 147
MCLO 109, 120, 130, **137**, 145
MCMF **116**
MCMM . 3, 5, 8, 9, 20, 21, 23, 28, 31, 32, 45, 51, 76, 80, 83, 84, 86, 87, 88, 89, 91, 93, **101**, 102, 103, 105, 106, 107, 108, 109, 111, 114, 115, 127, 139, 152, 153
MCNV 84, 102, 103, 108, **109**, 128, 137
MCOP 28, 29, 83, 84, 88, 89, 90, 103, **103**, 139
MCRC 6, 106, **106**
MCSD 109, 110, 118, 120, **127**, 130, 134, 136, 137, 138, 145, 146
MCSM 45, 76, **108**, 111, 114, 115, 116
MCSS 88, 89, 92, 101, 102, 103, **105**
MCTS **107**
MDAP **123**
MDAR **122**
MDAV **126**
MDBA **124**
MDDA **124**
MDDI **123**
MDFR **129**
MDFT **127**
MDIT 117, **126**
MDMC **127**
MDMT 119, **122**
MDRE 117, **129**
MDSA 118, **120**, 130, 137
MDVE 118, **129**
MDYN **117**, 120, 121, 122, 123, 124, 126, 127, 129, 130, 136, 137, 138
MHBD 120, **125**
MINI . 4, 28, 41, 42, 44, **52**, 53, 54, 56, 57, 60, 61, 62, 63, 75, 80, 81, 82, 83, 84, 93, 98, 99, 101, 103, 108, 109, 140, 142, 144, 145, 146
MMOD **32**
MNTA **170**
MOLS . 51, 87, 89, 91, 92, 101, 102, 103, 104, 108, 109, 112, **112**, 114, 127, 128, 129, 137, 144, 147
MSYM 5, **74**, 78, 79, 80, 84, 95, 172, 174
MTST **56**, 60, 146
MULT . 9, 21, 31, 32, 44, 52, 56, 60, **75**, 76, 80, 81, 83, 101, 115, 152
MWRT 4, **56**

N

NANT 79, **80**
NORE **81**
NPRC 8, 21, 22, 139, **152**
NSEQ 74, 79, **79**, 79, 172
NSRF 74, 79, **80**, 172
NSRO 74, 79, **79**, 172, 174

R

RCA4 ..... 51, 89, 91, 101, 103, 108, 111, 112, 113, **114**, 116, 127, 137
READ 16, **31**, 31, 32, 41, 44, 45, 49, 50, 52, 65, 68, 72, 76, 80, 81, 83, 84, 101, 103, 108, 129
REST **44**

RRHO **59**, 146
RWND 8, **44**, 83, 91, 141

S

SDLP 4, 52, **61**, 88
SEED **111**, 152
SMPL **116**
SOLV 2, 28, 34, 35, **39**, 41, 42, 80, 172
SPAT 9, **154**
SPMC ... 5, 8, 9, 20, 23, 83, 84, **102**, 103, 105, 109, 111, 112, 114, 153
SRNO **117**
SUBS 3, 32, 38, **64**, 66, 67, 72, 82, 84, 95, 129, 141, 144, 151, 172

T

TCMP 133, **134**
TIME 8, **150**
TOPN **32**, 32, 44
TORC 101, 108, **115**, 116
TORS . 51, 87, 89, 91, 92, 101, 102, 104, 107, 108, 109, 110, **111**, 112, 113, 114, 127, 128, 129, 144
TRED 18, **31**, 32, 44, 80, 81
TRES **112**

V

VBR2 ..... 4, **58**, 98, 99
VDWB 3, **51**, 51, 114, 141, 219
VIBR 4, **57**, 58

W

WRIT **31**, 32, 82, 118, 126, 129, 147

Z

ZMAT 109, 120, **134**

120 West 45th Street, 29th Floor
New York, NY 10036

101 SW Main Street, Suite 1300
Portland, OR 97204

8910 University Center Lane, Suite 270
San Diego, CA 92122

Zeppelinstraβe 13
81669 München, Germany

Dynamostraβe 13
68165 Mannheim, Germany

Quatro House, Frimley Road
Camberley GU16 7ER, United Kingdom

SCHRÖDINGER.