# SPASMS
## Documentation

## San Francisco Program of Applications
## for the
## Simulation of Molecular Systems

***Documentation Authors:***
**Allison Howard**
**Erik-Robert Evensen**
**Thomas Ferrin**
**David C. Spellmeyer**
**William C. Swope**
**David M. Ferguson**

***Software Authors:***
**David C. Spellmeyer**
**William C. Swope**
**Erik-Robert Evensen**
**Thomas Cheatham**
**David M. Ferguson**

***Major Contributors:***
**Shuichi Miyamoto**

**9 January 1995 Documentation Revision**

# TABLE OF CONTENTS

# PREFACE

This manual comprises the documentation for the SPASMS software. It is meant to introduce the user to algorithms used in the software, provide references for the calculational methods addressed by the software, aid the user in running the software, and to assist in the interpretation of output. It is not meant to be an exhaustive discourse on computational methods. The SPASMS documentation is organized into several sections:

a. *The Preliminaries* – This section of the documentation describes the SPASMS distribution software and testing suite. Instructions for the program installation are given and general descriptions of the input and output files to SPASMS are included.

b. *AMBER Force Field* – The Weiner *et al.* united- and all-atoms force fields (Weiner, S.J.; Kollman, P.A.; Case, D.A.; Singh, U.C.; Ghio, C.; Alagona, G.; Profeta, Jr., S.; Weiner, P. *J. Am. Chem. Soc.* **1984**, *106*: 765-784 and Weiner, S.J.; Kollman, P.A.; Nguyen, D.T.; Case, D.A. *J. Comput. Chem.* **1986**, *7*: 230-252) are used in SPASMS. The *AMBER force field* section describes these force fields. Particular attention is given to the specification of atom types used in the force fields.

c. *Minimization* – This section focuses on the optimization of potential energy with respect to Cartesian coordinates. The concepts of steepest-descent and conjugate-gradient minimization are introduced and two examples of minimization by SPASMS are discussed.

d. *Molecular Dynamics* – The *Molecular Dynamics* documentation introduces the user to a number of algorithms coded in SPASMS. The velocity Verlet, RATTLE, and temperature and pressure control algorithms, used in both molecular dynamics and Free Energy Perturbation, are described. In addition, the periodic boundary condition (PBC), switching function, and BUFFER region algorithms are discussed. These latter algorithms are used in minimization, molecular dynamics and Free Energy Perturbation. An example molecular dynamics run is also examined in this section.

e. *Free Energy Perturbation* – This section introduces the user to the concepts of free energy perturbation (FEP). It describes the thermodynamic perturbation algorithm used in SPASMS and other algorithms associated with this section of the program. A sample FEP calculation is also examined in the *Free Energy Perturbation* documentation.

f. *Detailed Description of SPASMS Input* – The control of the SPASMS program is given by keyword input. This documentation section gives a comprehensive description of the keywords. It also contains a flowchart to aid users in choosing keywords.

g. *SPASMS Quick Reference Guide* – The *SPASMS Quick Reference Guide* is intended for users who are already familiar with the SPASMS keyword input. The keywords are briefly described and their default values and data types are given in this portion.

We would appreciate receiving notification of all errors found in the documentation. We are also interested in any suggestions for improving the manual. Please include the ‘documentation revision date’, found on the front cover, in all mail related to this manual. The user is also encouraged to report all software bugs. When reporting a bug, please include the SPASMS version number (e.g., $Revision: 1.7 $ $Date: 91/03/05 17:40:55 $). This information may be found in subroutine *mdread* of the code and also in the MDOUT and MDDUMP output files. Electronic mail may be sent to ‘amber-request@cgl.ucsf.edu’ and surface/air mail should be addressed to SPASMS Software, Computer Graphics Laboratory, School of Pharmacy, University of California, San Francisco, CA 94143-0446.

Additional copies of this manual are available for $10 each, including postage, by writing to SPASMS Software Distribution, Computer Graphics Laboratory, School of Pharmacy, University of California, San Francisco, CA 94143-0446. Please include a check or money order payable to 'The Regents of the University of California' with your request. Sorry, but purchase orders cannot be accepted. Orders received without payment enclosed will be returned unprocessed.

The following article should be cited in all work in which the SPASMS software has been used:

> D. A. Pearlman, D. A. Case, J. W. Caldwell, W. S. Ross, T. E. Cheatham, S. DeBolt, D. M. Ferguson, G. Seibel, and P. A. Kollman, *Comput. Phys. Commun.* (1995) in press.

SPASMS was developed with support from the Defense Advanced Research Projects Agency under contract N00014-86-K-0757, the National Institutes of Health National Center for Research Resources grant RR-01081 and as part of a joint study project with IBM Corporation Palo Alto Scientific Center. We also thank Du Pont Merck Pharmaceuticals for their support.

## 1.1 THE SOFTWARE

The SPASMS software is a molecular mechanical minimization, molecular dynamics, and free energy perturbation program that incorporates many new features and advances in technology which allow users to study complicated dynamics problems. Except for the use of lower case in SPASMS code, the software is otherwise written in ANSI FORTRAN 77 and the code currently exceeds 16 000 lines. The software is written in a uniform and modular fashion so that users can make modifications easily. It is maintained at UCSF under the Revision Control System (RCS). This maintenance tool allows us to document all code changes and determine the differences between any two revisions of the code. All questions and comments regarding SPASMS should include the version number of the users' program (see *Preface*).

The major reference for the algorithms used in SPASMS is Allen, M. P.; Tildesley, D. J. *Computer Simulations of Liquids* (**1987**) Oxford University Press, Oxford. The user of SPASMS is strongly encouraged to read this text. Three other secondary references provide a good background for molecular mechanical simulations: 1) Burkert, U.; Allinger, N. L. *Molecular Mechanics* (**1982**) ACS Monograph *177*, American Chemical Society, Washington, D. C.; 2) McCammon, J. A.; Harvey, S. C. *Dynamics of Proteins and Nucleic Acids* (**1987**) Cambridge University Press, Cambridge; and 3) Brooks, C. L.; Karplus, M.; Pettitt, B. M. *Proteins: A Theoretical Perspective of Dynamics, Structure, and Thermodynamics* (**1988**) Advances in Chemical Physics, *71*, John Wiley & Sons, New York.

The authors of SPASMS have chosen to use the Weiner *et al.* molecular mechanical force field. Two key references to this force field are 1) Weiner, S.J.; Kollman, P.A.; Case, D.A.; Singh, U.C.; Ghio, C.; Alagona, G.; Profeta, Jr., S.; Weiner, P. *J. Am. Chem. Soc.* **1984**, *106*: 765-784 and 2) Weiner, S.J.; Kollman, P.A.; Nguyen, D.T.; Case, D.A. *J. Comput. Chem.* **1986**, *7*: 230-252.

Spasms code is largely machine-independent. One subroutine, *timer*, is machine dependent and the specific machine code is unmasked during the installation procedure. In addition, compiler directives have been added to the code in order to optimize program speed on specific hardware.

Machine and architecture dependencies are handled with the use of an *unmasking* procedure, whereby four-character tokens within comments in the code are recognized and translated appropriately for the target machine by the program *unmask* in the src/spasms/util/ directory. This method is more portable than the UNIX-based method used for the rest of the AMBER release (which can generate source for other operating systems but does not allow the other operating systems to generate source from the raw release), although it is more difficult for the programmer to expand.

The UNIX release features utilities for merging multiple source files into a single one and splitting the single file into multiple ones, for ease of moving to and from less flexible development environments.

## 1.2 INSTALLATION

The spasms distribution consists of a computer tape or ftp-able archive containing SPASMS and the file preparation programs PREP, LINK, EDIT and PARM, along with documentation and demos. If SPASMS is being provided independently of the complete AMBER release, various AMBER programs will be deleted, although traces of them may exist in the distribution tree.and printed documentation. The documentation contains *introductions* to the minimization, molecular dynamics, and free energy perturbation computations. It also describes the amber force field and provides both detailed and condensed descriptions of the spasms options.

The UNIX and VMS top directory is *amber41*, which includes a 0README file describing the release. Other 0README files may be found in various subdirectories. The doc subdirectory contains the postscript files necessary to generate additional copies of the SPASMS documentation. The *src/spasms* directory contains the split file of the SPASMS code, one file per routine. The single-file version can easily be created in UNIX by the command `make bigsource', although there is no need for this when compiling under UNIX unless debugging.

The *src/spasms/util* subdirectory contains the *unmask* and *fsplit* programs. A number of example calculations are contained in the *test2/spasms* directory. The primary purpose of this directory is as a test suite but the user may also find the files to be pedagogically useful tools.

See the 0README files in the top and *src* directories for the installation procedures. When you select a machine dependency file from *src/Machine/* and copy it to src/MACHINE, note the SPASMS configuration line for dimensioning: you may choose from two options allowing different approximate limits on simulation size: SMALL (less than 5000 atoms) or LARGE (less than 20000 atoms).

## 1.3 SPASMS TESTING SUITE

The testing suite in *test2/spasms/* contains input and output files of calculations designed to confirm the proper functioning of all major algorithms in SPASMS. It is strongly suggested that the user run these tests every time a change is made to the SPASMS code or whenever the code is ported to a machine. A listing of the directories in the testing suite and a description of their intent are given below. It is suggested the user read the *MDIN* and *0README* files for more detailed descriptions of the tests.

To run these tests on UNIX[a] machines, the user should change directories to *test2/spasms*. In order to run all the test cases, the command

```
Runtests
```

should be issued. If the user only wishes to run selected tests, the command

Test *directory_name*

should be entered. The *directory_name* option refers to the name of the directory for which a test run is desired (e.g., Test *peptide*).

*anisole:* The test cases in the anisole directory are devised to test the dihedral (*dan1sc and dan1st*), force routines as well as the RATTLE routines (in *vvlone* and *vvltwo*). An energy conservation test is used to determine if each of these routines is working properly. The parameter file for the anisole molecule includes an unusual dihedral angle with a periodicity of four, presented both as an example and as a test of the code for periodicity.

*butane:* The test cases in the butane directory are devised to test the bond (*bnd1st*), angle (*ang1st*), and dihedral angle (*dan1st*) force routines as well as the RATTLE routines (in *vvlone* and *vvltwo*). An energy conservation test as well as an RMS fluctuation test is used to determine if each of these routines is working properly.

*cap:* A test of the cap option may be performed using the files in this directory. The input and output files for a molecular dynamics simulation of methane in a ‘glob’ of water are provided.

*crambin:* The test cases in this directory are devised to test the nonbonded pair list generation routines, the nonbonded force routines and the routines for RATTLE.

*cramwat:* These test cases can be used to test several options. This is the only set of tests designed to examine the protein/water interface. The tests include energy conservation tests for *bnd1st, ang1st, dan1sc, dan1st, vvlone, vvltwo, setlep, setlev, nb1nex, nbexcl, ntip4p, wtip4p, wtip3p* and others. Nine different tests are included, some of which are extremely long runs. (**WARNING: THESE TESTS REQUIRE SIGNIFICANT AMOUNTS OF CPUTIME**)

*cyclobut:* Test case for the dihedral angle routines *dan1st and dan1sc*. The parm file includes a dihedral with a constraint angle (phase shift - 0¡ or 180¡).

*dihedral*: Tests are included for testing the dihedral driver option. Also included is a test to dump pdb files at the end of each pass through the dihedral driver's main loop.

*edyn*: A test of the ensemble dynamics routines in spasms. This example includes three tri-peptides all tied together at the $C\alpha$ coordinates. The "binding data" are contrived.

*gbutanes*: Tests of two different forms of cyclobutane. Usd to test the dihedral angle routines (*dan1st* and *dan1sc*).

*methane:* The test cases in this directory are devised to test the integration algorithm from a NVE ensemble of united-atom methane in water. Time step sizes of 20, 10, 5.0, 1.0, 0.5, and 0.1 psec are examined in the six input files and their corresponding output files. (**WARNING: TESTS METHANE5 AND METHANE6 REQUIRE SIGNIFICANT AMOUNTS OF CPUTIME**)

*multiple*: The input files for this test give examples of the use of multiple input and output files for one parm topology file. In addition, the ability to extract coordinates from a pdb file and the ability to write pdb files (with CONECT records) is tested. Also, a summary file is output, including the energy for each structure input and minimized.

*neon:* The input and output files for a free energy perturbation calculation are provided in this directory. The initial system of neon in 216 TIP4P waters is perturbed during a NPT molecular dynamics simulation.

*nh4:* In this free energy perturbation calculation, $NH_4^+$ is perturbed to $H_3O^+$ in the presence of three waters. The temperature is held constant during this simulation and periodic boundary conditions are not used.

*nmr*: A test of the nmr interatomic constraints and dihedral angle constraints.

*peptide:* A test of the belly option is given in this directory. The system consists of a hexapeptide and water cap. A molecular dynamics simulation is performed with an ‘infinite’ nonbonded cutoff and RATTLE on all bonds.

*settle_water*: A test of the analytical routine SETTLE, written by Shuichi Myiamoto. This routine is an analytical version of SHAKE/RATTLE that runs nearly 10 times as fast as the iterative methods. It only operates on water molecules. Three tests are included to ensure energy conservation. Each is a box of 216 TIP4P waters. Periodic boundaries are used with switching functions.

*water:* The test cases in the water directory are devised to test the nonbonded pair list generation routines and the nonbonded force routines. All of the tests use the same coordinate box of 216 water residues in a box 20• × 20• × 20• large.

## 1.4 PREPARING MOLECULES FOR SPASMS

Two or sometimes three external files are needed in order to run SPASMS. A parameter topology (PRMTOP) file and a coordinate (INPCRD) file contain AMBER force field parameters and the Cartesian coordinates of an ensemble of atoms, respectively. A third file, REFC, is sometimes used to restrain atoms to Cartesian positions. This file has the same format as the INPCRD file. The modules PREP, LINK, EDIT, and PARM need to be run to generate the parameter and coordinate files, or the LEaP program can be used to produce these two needed files directly. If the user decides to restrain atoms to the Cartesian positions of the INPCRD file, then the REFC file is generated as a copy of INPCRD.

## 1.5 SPASMS FILE DESCRIPTIONS

There are eleven files either read or generated by SPASMS.

| FORTRAN LOGICAL UNIT NAME | FORTRAN LOGICAL UNIT NUMBER | DEFAULT UNIT NAME | TYPE OF FILE | MVS, VM (IBM) DEFAULT FILE EXTENSIONS | UNIX[a]/VMS DEFAULT FILE EXTENSIONS |
|---|---|---|---|---|---|
| ioread | 5 | MDIN | input | INPUT | .in |
| istdot | 6 | – | – | – | – |
| iorest | 10 | RESTRT | output | RESTART | .restart |
| iorefc | 11 | REFC | input | REFC | .ref |
| iovel | 12 | MDVEL | output | MDVEL | .vel |
| ioenf | 13 | MDEN | output | MDEN | .en |
| iodump | 14 | MDDUMP | output | MDDUMP | .dump |
| iout | 16 | MDOUT | output | OUTPUT | .out |
| iocord | 17 | MDCRD | output | MDCRD | .crd |
| iostrt | 18 | INPCRD | input | START | .start |
| ioparm | 19 | PRMTOP | input | PARM | .parm |
| iodrv | 20 | MDDRV | input | none | none |
| ionmr | 21 | MDNMR | input | none | none |
| ioddmp | 22 | none | output | none | none |

INPCRD: The starting atomic Cartesian coordinates for minimization, molecular dynamics, or free energy perturbation calculations are read from this unit. In the case of an initial run, this file is created either by LEaP or by the PARM module of AMBER software. During a restart calculation, the RESTRT file from the previous run is used as the INPCRD file. In addition to atomic Cartesian coordinates, during restart runs (see RESTRT file) this file may also contain velocities, reference constraint positions, and periodic boundary boundary box dimensions (in that order).

MDCRD: Cartesian coordinates for atoms of a trajectory are output periodically to this unit during molecular dynamics or free energy perturbation calculations (see MDVEL and MDEN).

MDDRV: input file for the dihedral driver optional input parameters. Dihedral angles are specified by giving four atom numbers. The order is significant. However, the following convention gives the same dihedral angle if the order is reversed. I.E., the angle associated with atoms 19,33,13,11 is the same as that associated with atoms 11,13,33,19. The dihedral angle associated with atoms i,j,k, and l is determined by sighting along the j-k bond **FROM** atom j **TO** atom k. Imagine two vectors, the first points from atom j to atom i ($\mathbf{r}_{ij}$), and the second points from atom k to atom l ($\mathbf{r}_{lk}$). In this position, if the $\mathbf{r}_{ij}$ vector overlaps (is parallel to) the $\mathbf{r}_{lk}$ vector, the dihedral angle is zero. If the vectors are anti-parallel, the dihedral angle is 180¡. If $\mathbf{r}_{ij}$ must be rotated (about $\mathbf{r}_{kj}$) clockwise to overlap $\mathbf{r}_{lk}$, the dihedral angle is positive.

MDDUMP: During minimization, molecular dynamics, or free energy perturbation calculations, current diagnostic information is output through this unit. This information includes energy and timing data. The file is useful to gauge the progress of a calculation or to provide summary statistics concerning a calculation.

MDEN: Energy information may be output periodically to this unit during molecular dynamics or free energy perturbation calculations (see MDVEL and MDCRD). The title is first written to the file. Then, during each energy dump, the following information is written sequentially: MD step number, trajectory length, total energy, kinetic energy, potential energy, instantaneous temperature, VDW energy, hydrogen bond energy, electrostatic energy, bond energy, angle energy, dihedral energy, 1-4 VDW energy, 1-4 electrostatic energy, constraint energy, total virial, instantaneous volume, instantaneous pressure, and cap constraint energy. The user can investigate the format of this file in subroutine *endmp*. An example MDEN file is given below (please note that the format of the example file has been changed slightly, by reducing spacing, for the purposes of this documentation). The input for this file was the test case */tests/butane/but2.in* in which the keyword option ‘edumpfrequency = 500’ was added.

```
Butane test case No shake No Rattle 0.0010 ps timestep
500  0.50000      0.2438157222012502D+02
     0.1259726660227514D+02    0.1178430561784988D+02
352.1795
     0.33096435   0.00000000   0.00000000   1.82978352
     7.30184535
     1.29731509   1.02439731   0.00000000   0.00000000
     0.00000000
     0.00000000   0.00000000   0.00000000
1000 1.00000      0.2434653159642371D+02
     0.1004704903858861D+02    0.1429948255783510D+02
280.8835
     0.11742246   0.00000000   0.00000000   2.85931320
     7.84869066
     2.43314367   1.04091257   0.00000000   0.00000000
     0.00000000
     0.00000000   0.00000000   0.00000000
```

It is often useful to plot one or more of the energy components in this file *versus* time in order to examine the stability of a run or to search for conformational changes during the course of a simulation.

MDIN: This input file contains keywords which control the options and routines of SPASMS. An extensive description of this file is given in Sections 6 and 7 of the documentation.

MDOUT: Diagnostic information obtained throughout minimization, molecular dynamics, or free energy perturbation calculations is output through this unit. The output is discussed in detail in Sections 3, 4, and 5.

MDNMR: Input file for the nmr interatomic constraints and dihedral angle constraints. Only created and read if the flatwell option has been chosen.

MDVEL: Trajectory velocity coordinates are output periodically to this unit during molecular dynamics or free energy perturbation calculations. The frequency at which the coordinates are written is declared by the user in the MDIN file.

PRMTOP: This unit contains all of the molecular mechanical force field parameter information needed to initiate a minimization, molecular dynamics, or free energy perturbation calculation. The PRMTOP (parameter topology) file must be created either with the LEaP program or by the PARM module of AMBER software.

REFC: The REFC input file can be used to provide Cartesian coordinates to which the system is constrained during a minimization, molecular dynamics, or free energy perturbation calculation. There must be a one-to-one correspondence between atoms in the input coordinate (INPCRD) file and Cartesian positions for restraint in the REFC file. That is, a coordinate must be listed in REFC for each atom in INPCRD, even if it is unconstrained. Force constants for the restraining force are input through the MDIN file in *group input* format. Although the one-to-one correspondence between atoms in the INPCRD file and Cartesian positions in the REFC file must exist, only those atoms defined in the MDIN file *group input* will be restrained. If the user decides to restrain atoms to the Cartesian positions of the INPCRD file, then the REFC file is generated as a copy of INPCRD.

RESTRT: The *iorest* or "restart" output file minimally contains the Cartesian coordinates of a system at the end of a minimization, molecular dynamics, or free energy perturbation calculation. If a subsequent calculation is run, this file is then used as the input coordinate or INPCRD file. In addition to Cartesian coordinates, the file may also contain velocities, reference constraint positions, and periodic boundary box dimensions (in that order) when applicable.

*(istdot):* The *istdot* unit name is reserved for experimental code currently under development at UCSF.

*VM-CMS (IBM):* In order to run SPASMS under VM-CMS, the EXEC *spasms* is used. This EXEC is invoked by typing

spasms *parm start input*

where the three arguments are the file names of the parameter (PRMTOP), starting coordinate (INPCRD), and input option (MDIN) files, respectively. These files are expected to be formatted and of fixed length of 80 records. The files are also expected to have the following extensions PARM, START, and INPUT, respectively. The EXEC will cause the output files from the run to have the same file name as the input file and the file extensions listed below:

| | |
|---|---|
| INPUT | input file containing control data for the spasms run |
| MDCRD | output coordinate file containing saved dynamics positions |
| MDDUMP | debug "dump" file for monitoring execution status |
| MDEN | output file containing saved energy values |
| MDVEL | output file containing saved dynamics velocities |
| OUTPUT | output file to record results and diagnostics |
| PARM | input "parm" file containing molecular topology |
| REFC | input reference coordinates for positional constraints |
| RESTART | output restart file containing final positions and velocities |

| | |
|---|---|
| START | input “start” file with initial positions and velocities |

*UNIX[a]:* In the UNIX[a] environment, execution of SPASMS is controlled via a shell script command file. Since the main SPASMS executable program expects specific input and output file names, the shell script takes care of creating symbolic links to and from the user specified file names. After execution is completed, these symbolic links are removed. (Note that the use of `hard-wired’ link names means that only one job can be run in a directory at a time.) User filenames are specified on the input command line using the following flags:

```
spasms.csh  [ -i input ] [ -o output ] [ -p parm ] [ -c start ] [ -r restart ] [ -e energy ]
      [ -v velocity ] [ -d dump ] [ -x coord ] [ -z ref ] [ -s sum ] [-ri rootin ]
      [-ro rootout ] [ -ei extenin ] [ -eo extenot ] [-dd driverfile ] [ -nmr nmrin]
```

| | |
|---|---|
| **-i** | input file containing control data for the spasms run (default extension: ‘.in’) **always opened** |
| **-o** | output file to record results and diagnostics (default extension: ‘.out’) **always opened** |
| **-p** | input “parm” file containing molecular topology (default extension: ‘.parm’) **always opened** |
| **-c** | input “start” file with initial positions and velocities (default extension: ‘.start’) opened normally. Not opened if multiple input files requested. |
| **-r** | output restart file containing final positions and velocities (default extension: ‘.restart’) opened normally. Not opened if multiple output files requested. |
| **-e** | output file containing saved energy values (default extension: ‘.en’) |
| **-v** | output file containing saved dynamics velocities (default extension: ‘.vel’) |
| **-d** | debug “dump” file for monitoring execution status (default extension: ‘.dump’) |
| **-x** | output coordinate file containing saved dynamics positions (default extension: ‘.crd’) |
| **-z** | input reference coordinates for positional constraints (default extension: ‘.ref’) |
| -**s** | output file for the summary of multiple runs. (default extension: .sum) |
| -**ri** | the default name for multiple input file root names (default: SPIN) |
| -**ro** | the default name for multiple output file root names (default : SPOUT) |
| -**ei** | extension for multiple input files (default : .fmt) |
| -**eo** | extension for multiple output files (default: .fin) |
| -**dd** | dihedral driver input file (no default) |
| -**nmr** | nmr input file (no default) |

## 2.1 INTRODUCTION

Molecular mechanical force fields are based on the concept that the potential surfaces and physical properties of molecules may be calculated using classical mechanical functions. Hence, a molecule is thought of as a collection of atoms which interact with each other in a way that can be described by simple potential energy functions. This is called a force field. For example, one component of most force fields is the energy which arises from bond compression and stretching. This component is often approximated by a Hooke's law description.

$$U_{spring} = \frac{1}{2} K_r \left( r - r_0 \right)^2 \tag{2.1.1}$$

Thus, the bonding between two atoms is simulated in a manner analogous to two masses connected by a spring which is characterized by a natural force constant. Using this analogy, in Equation 2.1.1 the potential energy of the system of masses is given by $U_{spring}$ and the force constant of the spring is described by $K_r$. The equilibrium and displaced distances of the masses with respect to each other are described by $r_0$ and $r$, respectively. Both $K_r$ and $r_0$ are constants for each pair of masses connected by a certain spring. Likewise, in a molecular mechanical force field which uses a Hooke's law or harmonic description for the bonding between two atoms, there are constants which describe the force constant of a bond and the equilibrium bond length between the atoms. We call these constants *force field parameters* and they must be empirically derived from the physical properties of a prototypical set of molecules.

The molecular mechanical force field energy of a molecule is determined by the sum of the individual component functions of the potential, such as bond, angle, and electrostatic potentials (Equation 2.1.2). In turn, the energies of the individual force field components are a function of the deviation of the molecule from that of a hypothetical compound where the coordinates are at their equilibrium values.

$$E_{total} = \text{potential}_1 + \text{potential}_2 + \ldots + \text{potential}_n \tag{2.1.2}$$

The absolute energy of a molecule in molecular mechanics has no intrinsic physical meaning; the comparisons between molecular energies are important only in a relative sense. Energies calculated by molecular mechanics are related to the enthalpies of the molecules; however, they cannot be strictly considered enthalpies because thermal motion and hence, temperature dependent contributions, are excluded from the energy term.

Unlike quantum mechanical descriptions of molecules, electrons are not treated explicitly in most molecular mechanical calculations. For this reason, molecular mechanical calculations cannot generally be used to describe processes such as bond-making or bond-breaking or systems in which electronic delocalization or molecular orbital interactions play a major role in determining the geometry and properties.

The SPASMS package uses the AMBER force field for molecular mechanical calculations. The AMBER force field was originally developed to allow simulations of macromolecular systems. The user will discover that force field parameters are most highly developed for calculations of amino acids, peptides, proteins, and nucleic acids and their polymers. The force field potential, however, is certainly applicable to other molecules and the UCSF group is gradually refining parameters to address an even broader group of compounds.

## 2.2 A BRIEF HISTORY OF THE AMBER FORCE FIELD

The original AMBER force field was developed in order to provide an environment in which molecular mechanical calculations of biological and macromolecules could be conveniently carried out [Weiner, S.J.; Kollman, P.A.; Case, D.A.; Singh, U.C.; Ghio, C.; Alagona, G.; Profeta, Jr., S.; Weiner, P. *J. Am. Chem. Soc.* **1984**, *106*, 765-784]. Due to the restricted availability of computational resources, the force field was based upon the concept of *united-atoms*. In this force field, all hydrogens associated with methyl, methylene, or methine moieties were implicitly represented. The united-atom force field was developed for proteins and nucleic acids and was found to adequately simulate the structures and properties of biomolecules. The force field was soon extended to explicitly represent all of the atoms in a molecule [Weiner, S.J.; Kollman, P.A.; Nguyen, D.T.; Case, D.A. *J. Comput. Chem.* **1986**, *7*, 230-252] and this became known as the all-atom force field.

## 2.3 THE AMBER FORCE FIELD

AMBER is based upon a valence force field and the potential function is shown in Equation 2.3.1. The potential is represented by contributions from bond, simple angle, torsional, improper dihedral, van der Waals, hydrogen-bonding, electrostatic, and constraint functions. The individual force field functions are displayed graphically in Figure 2.3.2.

$$\begin{aligned} U_{total} = {} & \sum_{bonds} K_r \left( r - r_{eq} \right)^2 + \sum_{angles} K_\theta \left( \theta - \theta_{eq} \right)^2 + \\ & \sum_{dihedrals} \sum_{\eta} \frac{V_\eta}{2} \left[ 1 + \cos \left( \eta\phi - \gamma \right) \right] + \\ & \frac{1}{VDW_{scale}} \sum_{j=1}^{atoms} \sum_{i>j}^{atoms} \varepsilon_{ij}^{*} \left[ \left( \frac{R_{ij}^{*}}{r_{ij}} \right)^{12} - \left( \frac{R_{ij}^{*}}{r_{ij}} \right)^{6} \right] + \\ & \sum_{j=1}^{Hbonds} \sum_{i>j}^{Hbonds} \left( \frac{C_{ij}}{r_{ij}^{12}} - \frac{D_{ij}}{r_{ij}^{10}} \right) + \\ & \frac{1}{EEL_{scale}} \sum_{j=1}^{atoms} \sum_{i>j}^{atoms} \frac{q_i \, q_j}{\varepsilon \, r_{i\,j}} + \end{aligned} \tag{2.3.1}$$

$$\sum_{\text{constraints}} K_{\text{const}} \left( x - x_0 \right)^2 + \sum_{\text{cap atoms}} K_{\text{cap}} \left( y - y_0 \right)^2$$

Both the bond and angle potentials are in the form of simple harmonic functions. Hence, one would expect the (near) equilibrium bond and angle properties of a molecule to be represented well by the force field but the higher vibrational frequencies and bonds or angles far removed from equilibrium would be less well represented. The AMBER developers do not consider this to be a deficiency in the force field since most of the molecular systems examined with the force field are not highly strained and furthermore, the bond and angle contribution to the total energy is minimal for most of the molecules studied. Thus, these computationally simple functions which require few parameters to represent bonds and angles are used in preference to more complicated functions.

The dihedral potential function may be thought of as a truncated Fourier series. The AMBER force field will accept one, two, three, four, or six terms in the dihedral potential function. However, an examination of the AMBER force field parameters reveals that there are seldom more than two terms in the series and generally only one term. Here again, the developers have tried to assess the relative importance of this potential to the total energy of the molecules examined, and have found that the error in truncating the Fourier series after the first or second term is relatively insignificant as compared other approximations intrinsic to the force field. Using few terms in the Fourier series results in decreased computational time and a somewhat reduced period of time necessary to develop force field parameters. In Figure 2.3.1, a dihedral potential function is shown. Phi ($\phi$) is the particular dihedral angle for which the potential is calculated. The periodicity of the function is represented by eta ($\eta$) and the phase is represented by gamma ($\gamma$). In the AMBER force field, the phase can have only two values: 0 or 180 degrees. The power of this function can be ascertained from Figure 2.3.1, in which a the relative dihedral energies for a *cis* and *trans* isomer can be accurately simulated.

Improper dihedral angles, that is, dihedral angles in which the atoms are not sequentially bonded to each other, use the same potential function in the AMBER force field as torsional angles. Improper dihedral angles are used to preserve the planarity of specific ring fragments and to prevent racemization of chiral centers during minimizations and molecular dynamical simulations.

While the bond, angle, and dihedral potentials may be thought of as bonding interactions, the van der Waals, hydrogen-bonding, and electrostatic potentials describe the nonbonded interactions. Nonbonded potentials are calculated in AMBER only for those pairs of atoms which are separated by three or more bonds. This is due to the fact that the parameters which are derived for the bonds and angles incorporate 1-3 and closer nonbonded interactions.

A 6-12 function is used to simulate the van der Waals interactions. The attractive London dispersion interaction between two atoms is described by a $r^{-6}$ term and the repulsive interaction caused by Pauli exclusion is given by a $r^{-12}$ term. As pointed out by Weiner, *et al.* [Weiner, S.J.; Kollman, P.A.; Case, D.A.; Singh, U.C.; Ghio, C.; Alagona, G.; Profeta, Jr., S.; Weiner, P. *J. Am. Chem. Soc.* (**1984**) *106*, 765-784], the 6-12 function is not as physically appropriate as a 6-exponential function, but it is computationally simpler. It should be noted that 1-4 van der Waals and 1-4 electrostatic interactions are scaled, typically by a factor of 0.5, when evaluating the potential function (i.e., $VDW_{scale}$ and $EEL_{scale} = 2$). The rational given [Weiner, S.J.; Kollman, P.A.; Case, D.A.; Singh, U.C.; Ghio, C.; Alagona, G.; Profeta, Jr., S.; Weiner, P. *J. Am. Chem. Soc.* (**1984**) *106*, 765-784] for scaling the 1-4 van der Waals parameters is based largely on the fact that the 6-12 function is too steep for close interactions, which would be largely reflected in the 1-4 interactions, and also to compensate for the fact that the charge redistribution during a pairwise interaction, which would normally be more important for 1-4 interactions, is neglected in the molecular mechanics force field.

In the AMBER force field, the van der Waals function is *replaced* by a 10-12 function for those pairs of atoms which can participate in hydrogen-bonding. The hydrogen-bonding potential is used not so much to contribute to the hydrogen-bonding attraction between atoms, rather, it is implemented to *fine tune* the distances between these atoms.

Another term in the AMBER force field equation is the electrostatic function. The monopole (point) charges ($q_i$ and $q_j$ of Equation 2.3.1) are centered on each atom and are derived by fitting to a quantum mechanical electrostatic potential [Singh, U.C.; Kollman, P.A. *J. Comput. Chem.* (**1984**) *5*, 129]. Two forms of the electrostatic function may currently be chosen. For *in vacuo* simulations or simulations in which explicit water molecules are represented, a classical electrostatic function in which the denominator equals $eR_{ij}$ is usually chosen. In order to implicitly represent solvent, one typically uses a distance-dependent dielectric, where the denominator is equal to $eR_{ij} \bullet R_{ij}$. The effect of a distance-dependent dielectric is to reduce the magnitude of the long range electrostatic interactions as compared to the constant dielectric. In addition, the increased magnitude of the short range electrostatic interactions using a distance-dependent dielectric mimics the (usually neglected) effects of atomic polarization.

The final two terms in the AMBER force field equation are used to harmonically restrain Cartesian positions. The first of these is a constraint to reference coordinates. In this situation, the Cartesian coordinates of a system are restrained to lie near those of a reference system. The restraint is often used during minimization procedures so that the minimized structure of a compound will deviate only minimally from that of a reference – usually a experimentally derived structure. The cap option is used to restrain solvent molecules spatially. This potential is often used in calculations where CPU considerations limit the number of solvent molecules which can be used, and the scientist solvates only a portion of a molecule, such as an enzyme's active site. When the cap option is used, the solvent molecules will not drift appreciably during the course of the simulation.

## 2.4 AMBER ATOM TYPES

As mentioned previously, the atom types for which the AMBER force field is parameterized fall into several categories: united- and all-atom representations of carbon atoms and other atom types. In the united-atom model, methyl, methylene, and methine functional groups are represented by only one atom. Hence, a methyl group would be described by one atom, not four, and that atom would be characterized by a larger than typical van der Waals radius and mass. The all-atom model explicitly represents each hydrogen atom attached to a carbon. The obvious reason for developing the united-atom types is to reduce computational time. The computational time in molecular mechanics calculations varies as a function of $n^2$, where $n$ is the number of atoms in the system.

There are several ways to utilize the different atom representations in AMBER. It is possible to create molecules which contain exclusively united-atom or all-atom carbons. Generally, most small molecules are described by the all-atom model as this offers a somewhat better description of the potential surface. Macromolecules are frequently described by a mixture of the two representations: those portions of the molecule which are most important for biological activity are formed from the all-atom types and the regions in which it is less necessary to describe to a high accuracy are modeled using united-atoms.

The specific atom types which have been parameterized for the AMBER force field are listed below with a short description of their appropriate use [Weiner, S.J.; Kollman, P.A.; Case, D.A.; Singh, U.C.; Ghio, C.; Alagona, G.; Profeta, Jr., S.; Weiner, P. *J. Am. Chem. Soc.* (**1984**) *106*, 765-784. Weiner, S.J.; Kollman, P.A.; Nguyen, D.T.; Case, D.A. *J. Comput. Chem.* (**1986**) *7*, 230-252]. These atom types are also displayed pictorially in Figure 2.4.3.

**Carbons**

*United-Atom Types*

| | |
|---|---|
| • C2 | $sp^3$ carbon with two implicit hydrogens |
| • C3 | $sp^3$ carbon with three implicit hydrogens |
| • CD | $sp^2$ aromatic carbon, with one implicit hydrogen, in a six-membered ring |
| • CE | $sp^2$ aromatic carbon, with one implicit hydrogen, in a five-membered ring; the carbon is bonded to two nitrogens, as in purines |
| • CF | $sp^2$ aromatic carbon, with one implicit hydrogen and next to a nitrogen without a hydrogen, in a five-membered ring (e.g., $C_{\delta}$-$N_{\varepsilon}$=$C_{\varepsilon}$ in histidine) |
| • CG | $sp^2$ aromatic carbon, with one implicit hydrogen and next to a NH, in a five-membered ring (e.g., $C_{\delta}$-$N_{\varepsilon}$–$C_{\varepsilon}$ in histidine) |
| • CH | $sp^3$ carbon with one implicit hydrogen |
| • CI | $sp^2$ aromatic carbon, with one implicit hydrogen and between two *NC* nitrogens, in a six-membered ring of purines |
| • CJ | $sp^2$ carbon, with one implicit hydrogen, at positions 5 and 6 in pyrimidines (more double bond than aromatic character) |
| • CP | $sp^2$ aromatic carbon, with one implicit hydrogen and between two nitrogens, in a five-membered ring (e.g., in histidine) |

*All-Atom Types*

| | |
|---|---|
| • C | $sp^2$ carbonyl and aromatic carbon, with hydroxyl substituent, in tyrosine |
| • C* | $sp^2$ aromatic carbon, with one substituent, in a five-membered ring (e.g., $CE_{\gamma}$ in tryptophan) |
| • CA | $sp^2$ aromatic carbon, with one substituent, in a six-membered ring |
| • CB | $sp^2$ aromatic carbon at junction between five- and six-membered rings (e.g., $CE_{\delta}$ in tryptophan, C4 andC5 in purines) |

| | |
|---|---|
| CC | sp$^2$ aromatic carbon, with one substituent and next to a nitrogen, in a five-membered ring (e.g., CE$_\gamma$ in histidine) |
| CK | sp$^2$ aromatic carbon, between two nitrogens and bonded to one explicit hydrogen, in a five-membered ring (in purines) |
| CM | sp$^2$ carbon, with one substituent, at positions 5 and 6 in pyrimidines (more double bond than aromatic character) |
| CN | sp$^2$ aromatic junction carbon in between five- and six-membered rings (e.g., CE$_\varepsilon$ in tryptophan) |
| CQ | sp$^2$ aromatic carbon, between two *NC* nitrogens and bonded to one explicit hydrogen, in a six-membered ring of purines |
| CR | sp$^2$ aromatic carbon, with one explicit hydrogen and between two nitrogens, in a five-membered ring (e.g., in histidine) |
| CT | sp$^3$ carbon with four explicit substituents |
| CV | sp$^2$ aromatic carbon, next to a nitrogen without a hydrogen and bonded to one explicit hydrogen, in a five-membered ring (e.g., C$_\delta$-N$_\varepsilon$=C$_\varepsilon$ in histidine) |
| CW | sp$^2$ aromatic carbon, next to a *NH* and bonded to one explicit hydrogen, in a five-membered ring (e.g., C$_\delta$-N$_\varepsilon$-C$_\varepsilon$ in histidine) |

### Hydrogens

| | |
|---|---|
| H | amide and imino hydrogens |
| H2 | amino hydrogens from NH$_2$ in purines and pyrimidines |
| H3 | positively charged hydrogens of lysine and arginine |
| HC | explicit hydrogen attached to carbon |
| HO | hydrogen on hydroxyl or water oxygen |
| HS | hydrogen attached to sulfur |
| HW | hydrogen attached to TIP3P water model oxygen atom |

### Lone Pair

| | |
|---|---|
| LP | lone pairs |

### Nitrogens

| | |
|---|---|
| N | sp$^2$ nitrogen in amide groups |
| N* | sp$^2$ nitrogen, with attached alkyl group (N9 in purines, N1 in pyrimidines), in purines and pyrimidines |
| N2 | sp$^2$ nitrogen in nucleic acid amino groups and arginine amino group |
| N3 | sp$^3$ nitrogen with four substituents (e.g., lysine-N$_\zeta$) |
| NA | sp$^2$ nitrogen, with attached hydrogen, in five-membered ring (e.g., protonated histidine) |
| NB | sp$^2$ nitrogen, with lone pairs, in five-membered ring (e.g., N7 in purines) |
| NC | sp$^2$ nitrogen, with lone pairs, in six-membered ring (e.g., N3 in adenine) |
| NT | sp$^3$ nitrogen with three substituents (e.g., unprotonated amines) |

### Oxygens

| | |
|---|---|
| O | carbonyl oxygen |

- O2 carboxyl and phosphate nonbonded oxygens
- OH alcohol oxygen
- OS ether or ester oxygen
- OW oxygen in a TIP3P water model

**Phosphorus**

- P phosphorus in phosphate groups

**Sulfurs**

- S sulfurs in disulfide linkages
- SH sulfur in cystine

# SECTION 3: MINIMIZATION

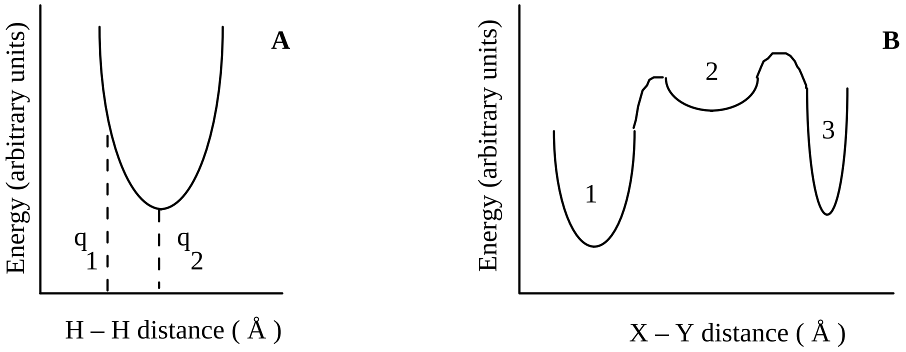
## 3.1 OVERVIEW

A widely used technique in computational chemistry is that of energy minimization. Using this method, one tries to find the coordinates that represent the minimum potential energy of a system. That is, for the potential energy function *U* (Equation 2.3.1), and Cartesian coordinates **q**, one would like to find a value such that

$$\frac{\partial U}{\partial \mathbf{q}_{\mathrm{i}}} = 0 \tag{3.1.1}$$

There are three frequent goals of minimization: 1) to optimize the geometry of a system, 2) to calculate the relative energies for the equilibrium structures of several different species, and 3) to remove large forces on atoms in preparation for a molecular dynamics calculation.

(3.1.2)



In Figure 3.1.2a, a geometry optimization of the diatomic hydrogen is depicted. In this one-dimensional illustration, the initial H – H bond distance ($q_1$) is shorter than the equilibrium distance. Minimization will bring this distance to the potential minimum at $q_2$ where the slope of the objective function is zero. Unfortunately, the situation for most molecules is much more complex. Molecules typically have many degrees of freedom and this will result in numerous minima on the potential energy surface. Finding the global minimum, that is, the minimum representing the most stable configuration of atoms, in still an insurmountable problem for a molecule containing more than about 10 dihedral angles. If the initial structure of a molecule lies near the well '3' of Figure 3.1.2b then minimization will not find the global minimum at '1'. Rather, the equilibrium structure will represent that at the bottom of the well in '3'. If the molecule contains few independent variables, it would be possible to start the minimization using many different initial structures with the possible result that one structure would lie close enough to the global minimum to find the most stable configuration of atoms during geometry optimization.

One point to keep in mind regarding minimizations of molecules is that the final structure may be of limited use in solving chemical problems. Since temperature is excluded from minimizations, the entropic contributions to the stability of a molecule are not included. Furthermore, molecules do not exist as static structures: they are dynamic and the "natural structure" of a molecule represents an average sampling of the potential surface.

The starting coordinates for many minimization calculations are that of an experimental structure.  X-ray structures are frequently used for proteins and microwave or electron diffraction structures are useful for smaller systems. Structures from theoretical calculations are also frequently used for the initial coordinates.  Two of these more popular calculations are quantum mechanics and distance geometry.  Sometimes, part of an input structure is better determined than other parts.  For example, the backbone atoms in an X-ray structure of a protein may be better resolved than the sidechain atoms.  It is useful in cases such as these to make several minimization runs.  Positional constraints would be associated with the backbone atoms and the constraint force constant would be diminished with each successive run.  This results in a minimized structure having an RMS deviation from the initial structure that is less than it would be otherwise.

Having a minimization calculation converge does not necessarily mean that a minimum in the potential surface has been found.  In Figure 3.1.2b, a stationary point is depicted connecting the potential wells ‘2’ and ‘3’.  In this stationary point the potential energy surface is a maximum.  If a first derivative minimizer is used, the single criterion of slope guides the minimizer and the slope will be zero for both a maximum stationary point and a potential minimum.  In order to differentiate between between the two, the eigenvalues of the second derivative matrix of the energy with respect to positional coordinates must be calculated, as in a normal mode calculation.  A true minimum will have only six such eigenvalues equal to zero.

## 3.2 STEEPEST-DESCENT METHOD

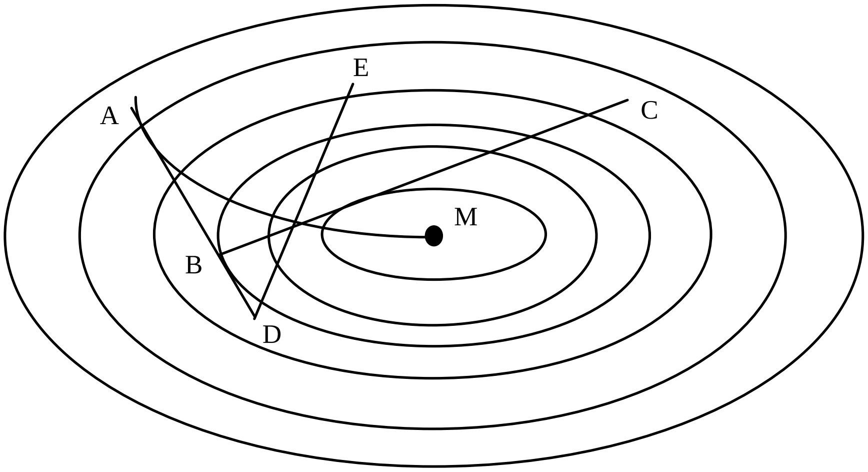
The steepest-descent method is a first order minimizer.  That is, it makes use of the first-derivative of the potential energy with respect to the Cartesian coordinates.  The descent is accomplished by adding an increment to the coordinates in the direction of the force or the negative gradient of the potential energy.  This can be represented by Equation 3.2.1

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \lambda \mathbf{s}_k ; \quad \mathbf{s}_k = \frac{\mathbf{F}}{|\mathbf{F}|} ; \quad \mathbf{F} = -\nabla U \tag{3.2.1}$$

where **x** are the Cartesian vectors, $\lambda$ is the variable increment or step size, and **s** is the search vector.

Figure 3.2.2 represents a potential energy surface with a minimum at ‘M’.  If the minimization begins at point ‘A’ and proceeds with infinitesimally small $\lambda$, the path ‘A – M’ will be taken during steepest-descent minimization.  If $\lambda$ was increased and the first step was ‘A – B’, then the subsequent step would proceed along ‘B – C’. Finally, a large initial $\lambda$ might initially place the system at ‘D’.  A second step would be along path ‘D – E’ which is diverging from the minimum.

(3.2.2)



The steepest-descent method is notable for its rapid alleviation of large forces on atoms. That makes the technique especially useful for eliminating the large non-bonded interactions often found in initial structures. Each step in steepest-descent requires a minimal amount of CPU time. Also, since the method is not able to surmount potential barriers, the final structure will lie structurally close to the initial geometry. However, its disadvantage is that it converges very slowly toward a minimum. Therefore, people often use this method for the first few steps of a minimization and then switch to a more efficient method.

## 3.3 CONJUGATE-GRADIENT METHOD

SPASMS uses a conjugate-gradient method for energy minimization. The original code and algorithm were written by Shanno and Phua (Shanno, D. F.; Phua, K. H. *ACM Trans. On Mathematical Software* (**1980**) *6*: 618-622). The code was extensively modified by William Swope following the work of Watowich *et al.* (Watowich, S. J.; Meyer, E. S.; Hagstrom, R.; Josephs, R. *J. Comp. Chem.* (**1988**) *9*: 650). It differs from the steepest-descent technique in that both the current gradient and previous search direction are used to drive the minimization. The conjugate-gradient method is also a first order minimizer. For the conjugate-gradient method

$$\mathbf{s}_k = -\mathbf{g}_k + b_k \mathbf{s}_{k-1} \; ; \; b_k = \frac{\left| \mathbf{g}_k \right|^2}{\left| \mathbf{g}_{k-1} \right|^2} \tag{3.3.1}$$

where **s** is again the search vector, **g** is the gradient of the energy, and b is a scaling factor of the current and previous gradients. The advantage of the conjugate-gradient minimizer is that it converges faster than the steepest-descent technique since it uses the minimization history to calculate the current search direction. It contains a scaling factor, $b_k$, for determining the step size and therefore, as compared to the steepest-descent method, the step sizes are more optimal.

(3.3.2)

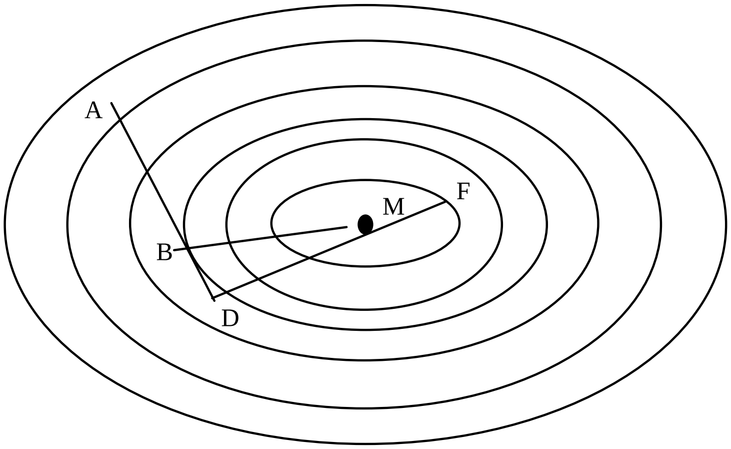


Figure 3.3.2 shows the same potential as in Figure 3.2.2 and the steps of a conjugate-gradient minimization can be compared with the steepest-descent method. Here, a system will be at the potential minimum after the second step if the first step proceeds from point ‘A’ to point ‘B’. If the first step is too great, bringing the system to point ‘D’, the second step will still place the system near the minimum because of the knowledge regarding the the penultimate step.

```
Title = " Butane minimzation "
!
! Run a minimization to an rms of 0.000001
! or a maximum of 5000 steps
!
   minimization
   convergencecriteria = 0.000001
   steps = 5000

   timelimit = 800.0
   coordinateinput = crd
   prtfreq = 10
   nbupdate = 9999
   bufferupdate = 1001
   pairlist = atom
   cutoff = 99.0
   dielectric = 1.0
   velreassign = -1
   firstwater = 99999
```

## 4.1 INTRODUCTION

In molecular dynamics, we would like to calculate the positions and velocities that particles will have in the future, based on their current positions and velocities. This is done by first determining the force on each particle ($\mathbf{F}_i$) as a function of time, which is equal to the negative gradient of the potential energy (Equation 4.1.1):

$$\mathbf{F}_i = -\frac{\partial U}{\partial \mathbf{r}_i} \tag{4.1.1}$$

where $U$ = potential energy function
$\mathbf{r}$ = position of a particle

The acceleration, **a**, of each particle can then be determined by dividing the force acting on it by the mass of the particle (Equation 4.1.2):

$$\mathbf{a}_i = \frac{\mathbf{F}_i}{m_i} \tag{4.1.2}$$

The change in velocities is equal to the integral of acceleration over time and the change in position is equal to the integral of velocity over time (Equation 4.1.3):

$$d\mathbf{v} = \int \mathbf{a}\,dt \ ; \quad d\mathbf{r} = \int \mathbf{v}\,dt \tag{4.1.3}$$

Finally, the kinetic energy can be defined in terms of of both the velocities and momenta of the particles:

$$K(\mathbf{v}) = \frac{1}{2}\sum_{i=1}^{N} m_i \mathbf{v}_i^2 \ ; \quad K(\mathbf{p}) = \frac{1}{2}\sum_{i=1}^{N} \frac{\mathbf{p}_i^2}{m_i} \tag{4.1.4}$$

The total energy of the system, called the Hamiltonian, is the sum of the kinetic and potential energies:

$$H(\mathbf{q},\mathbf{p}) = K(\mathbf{p}) + U(\mathbf{q}) \tag{4.1.5}$$

where $\mathbf{q}$ = the set of Cartesian coordinates
$\mathbf{p}$ = the momenta of the particles

where the AMBER energy function (equation 2.3.1) represents the potential energy $U(\mathbf{q})$.

We express the atomic positions at a particular time, t, $\mathbf{q}_i(t)$. The velocities, $\mathbf{v}_i(t)$, are the first derivative of the positions with respect to time:

$$\mathbf{v}_i(t) = \frac{d}{dt}\mathbf{q}_i(t) \tag{4.2.1}$$

SPASMS employs the velocity Verlet algorithm of Swope and Andersen to integrate the equations of motion. The positions are updated by:

$$\mathbf{q}_i(t + \delta t) = \mathbf{q}_i(t) + \delta t\mathbf{v}_i(t) + \frac{\delta t^2}{2m_i} \tag{4.2.2}$$

and the velocities for the next time step are calculated by:

$$\mathbf{v}_i(t + \delta t) = \mathbf{v}_i(t) + \frac{\delta t}{2m_i}\left[\mathbf{F}_i(t + \delta t) + \mathbf{F}_i(t)\right] \tag{4.2.3}$$

This technique involves updating the velocities in two steps. In the first step, the velocities are advanced from time (t) to time (t + δt/2). Then forces (**F**) are updated for the new atomic positions and the velocities are updated from time (t + δt/2) to time (t + δt). The resulting set of positions and velocities, $\mathbf{R} = \{\mathbf{q}(t_1), \mathbf{q}(t_1),\ldots,\mathbf{v}(t_1), \mathbf{v}(t_2),\ldots\}$, is referred to as the dynamics trajectory.

In contrast to the leap frog integration method, the velocity Verlet algorithm allows both the positions and velocities of the particles to be known at the same time step which has several distinct advantages. The most important benefit is that the velocities can be manipulated in a straight forward manner, making it possible to utilize several different methods for coupling to temperature or pressure baths. Another major advantage to this scheme is that the time step can be altered between runs without a deleterious effect on the system. This is not true of the leap-frog method. In addition, because the positions and velocities are independent of the previous time steps the velocity Verlet method allows for complete determination of the system at every time step.

The equations of motion are “integrated” through the use of very small time steps. At each time step, the energy and forces of the system as well as any constraints, such as RATTLE or SETTLE, are evaluated. The velocities are updated, and new positions are calculated. The time step for the evolution of molecular dynamics is very small – usually on the order of 0.5-1.5 femtoseconds. This is a result of the need to adequately integrate the high frequency motions of the system. Since these are usually the bonds, which vibrate on the order of picoseconds, the time step size must be small enough to accurately simulate the bond vibrations. The next highest frequency motions are typically angles. Clearly then, one disadvantage of this type of computation is that the trajectory length which one can calculate will be several orders of magnitude shorter than any chemical process and most physical processes, which occur on nanosecond and larger time scales.

Integration of the equations of motion with the Hamiltonian shown above (equation 4.1.5) is referred to as a classical molecular dynamics simulation. In a classical simulation, there are no external sources (or depositories) of energy. That is, there are no other energy terms in the Hamiltonian. Thus, the total energy of the system should be conserved. A sensitive test case for dynamics programs entails determining the change in kinetic energy and the potential energy between time steps. In the classical (microcanonical) ensemble, the change in kinetic energy should be of opposite sign and of exact magnitude of the change in potential energy. The total energy of the system should not change during the simulation; naturally, this is difficult to achieve on machines with finite precision.

Frequently it is not feasible to run a constant energy molecular dynamics simulation. The major reason for this is that we are limited by the amount of computer time that we have available. Several approximations to the energy term (usually the potential energy) are made. This requires that the Hamiltonian be modified somewhat to correct for the approximations. The most common modifications include the use of temperature scaling (changing the velocities so that a desired amount of kinetic energy is maintained) and pressure scaling (changing the positions so that a desired pressure is maintained).

The most extreme approximation to the potential energy is that not all intermolecular forces are calculated. Specifically, the Lennard-Jones and charge-charge interactions at long distances are ignored. The distance at which these interactions are no longer calculated is referred to as the “cutoff” distance. Preferably, the cutoff distance is chosen so that the potential energy at the cutoff and beyond is nearly zero. This is not practical in most cases. Typically, when one uses a short cutoff distance, the system will increase in temperature (an increase in kinetic energy). This is a result of the exchange of the potential and kinetic energy in the system. The common “fix” for this is the use of a temperature coupling scheme where one removes the kinetic energy at specific points in the trajectory. The exact method will be outlined below. Another solution to this problem is the application of a smoothed potential; i.e., applying the switching function (equations 4.2.4 - 4.2.6). The switching function is applied by scaling the nonbonded terms (Lennard-Jones, coulombic, and hydrogen bonds) by the following equation:

$$S(r) = c_0 + c_1\left[\frac{r - r_{lower}}{r_{upper} - r_{lower}}\right] + c_2\left[\frac{r - r_{lower}}{r_{upper} - r_{lower}}\right]^2 + c_3\left[\frac{r - r_{lower}}{r_{upper} - r_{lower}}\right]^3 + c_4\left[\frac{r - r_{lower}}{r_{upper} - r_{lower}}\right]^4 + c_5\left[\frac{r - r_{lower}}{r_{upper} - r_{lower}}\right]^5 \tag{4.2.4}$$

Where, the coefficients are determined so that the following conditions are met:

(4.2.5)

$$S\Big|_{r = r_{lower}} = 1; \quad \frac{dS}{dr}\Big|_{r = r_{lower}} = 0; \quad \frac{d^2S}{dr^2}\Big|_{r = r_{lower}} = 0;$$

$$S\Big|_{r = r_{upper}} = 0; \quad \frac{dS}{dr}\Big|_{r = r_{upper}} = 0; \quad \frac{d^2S}{dr^2}\Big|_{r = r_{upper}} = 0$$

These conditions guarantee that the potential function is unaltered inside of the lower switch distance, that the function goes to zero smoothly, and that the function is thrice differentiable, a requirement for all potentials used when integrating the equations with a derivative of the Verlet method. The coefficients which meet these criteria are:

$$c_0 = 0; \quad c_1 = 0; \quad c_2 = 0; \quad c_3 = -10; \quad c_4 = 15; \quad c_5 = -6 \tag{4.2.6}$$

## 4.3 ENSEMBLES

Any of several possible conditions or ensembles may be chosen in which to run an MD simulation. To monitor conformational changes in a molecule it may be adequate to run an *in vacuo* simulation or a simulation in which only a solvent shell is applied. Although neither system is physically rigorous, each requires relatively little CPU time and may be sufficiently accurate to provide a basic description of the structural transitions within a molecule.

Sometimes it is desirable to calculate thermodynamic quantities associated with a collection of atoms. In these cases, it is necessary to employ somewhat more elaborate schemes to fix the conditions of the calculation, such as the total energy (E), number of particles (N), volume (V), temperature (T), or pressure (P) in the system. Three standard ensembles used in molecular dynamics are the microcanonical (NVE), isobaric-isothermal (NPT) and canonical (NVT) ensembles.

The NPT and NVT ensembles are frequently used in SPASMS. The system of interest is placed in a solvent box and periodic boundary conditions (see Section 4.6) are applied. A simulated coupling to heat and pressure baths in the NPT ensemble is then made. The microcanonical (NVE) ensemble, however, is seldom used in AMBER, but should be applied frequently in SPASMS. In theory, the total energy of a system is conserved during molecular dynamics if the integration of the equations of motion is accurate. In practice, the total energy is not usually conserved while running utilizing a nonbonded cutoff without switching functions. This results in a continual increase in total energy during the course of the simulation. SPASMS, however, makes use of switching functions to eliminate problems with the nonbonded cutoff.

## 4.4 TEMPERATURE AND PRESSURE CONTROL

In order to compare calculations with experimental measurements, it is often useful to allow the computational system to exchange energy with its surroundings under the conditions of constant temperature and pressure (NPT and NVT ensembles). That is, we add a term to the Hamiltonian that constrains the temperature to a desired value. During the simulation, the velocities are adjusted so that the temperature of the system is near the desired value. In SPASMS, the procedures of Berendsen *et al.* [Berendsen, H.J.C.; Postma, J.P.M.; Gunsteren, W.F.; DiNola, A.; Haak, J.R. (**1984**) *Journal of Chemical Physics*, *81*, 3684] and Anderson [Andersen, H. C. (**1980**) *Journal of Chemical Physics*, *72*, 2384] are available to enforce the constant temperature condition.

During constant temperature molecular dynamics, the velocities ($v$) in the system are scaled during each time step of MD. This couples the system, with a temperature relaxation time of $\tau_T$, to a heat bath at $T_0$. Both $T_0$ and $\tau_T$ are assigned in the SPASMS' control input file. The velocities are scaled by a factor $\lambda v$, where

$$\lambda = 1 + \frac{\delta t}{2\tau_T}\left(\frac{T_0}{T - 1}\right) \tag{4.4.1}$$

A feature of SPASMS is that it will scale the velocities exactly to the desired temperature value, if so desired. This can be useful when the temperature fluctuations are very large and/or the instantaneous temperature of the system is very different from the desired value. Use of this temperature scaling with no changes in volume is similar to an NVT simulation. It should be noted that this method does not exactly correspond to the NVT ensemble. Because the velocities are being scaled based on the kinetic energy at every time step, the kinetic energy is not allowed to fluctuate within what would be a reasonable range. Thus, if the coupling parameter $T$ is too "tight" one obtains an iso-kinetic energy ensemble, rather than an isothermal ensemble. This is highly undesirable because the system will not average the phase space about a desired distribution of kinetic energies.

Two other methods of temperature control are available in SPASMS. [Andersen, H. C. (**1980**) *Journal of Chemical Physics, 72*, 2384]. These can be used in conjunction with the previously described method, or independently. Both of these methods derive from Andersen's method of stochastic collisions. The first scheme is called MAJOR COLLISIONS and the second is MINOR COLLISIONS. In major collisions, every atom in the system is reassigned a randomly chosen velocity from a Boltzman distribution about a given temperature. The entire system is given a new set of randomly chosen velocities between time steps. In minor collisions, a specified number of atoms are given randomly chosen velocities from a distribution about the desired temperature. The number of atoms to be reassigned is specified by the user. The specific atoms are chosen at random based on a uniform random distribution. These atoms are then given new velocities and the simulation proceeds. In both cases, RATTLE is done immediately after the reassignment to eliminate intramolecular velocity components along the bonds. In a system with $N$ atoms, it is apparent that MAJOR collisions and MINOR collisions are the same procedure if one chooses to reassign all $N$ velocities with the MINOR collision method.

One can choose any one of these methods, or one can combine the two different types of stochastic coupling during a given simulation. For instance, one might choose to scale the velocities every time step with a coupling parameter of 0.4 $\text{psec}^{-1}$ and scaled exactly every 1000 time steps, have major collisions every 100 steps, and minor collisions with 12 atoms every time step.

In a manner similar to the velocity scaling, the pressure is held constant by rescaling the atomic coordinates by a factor $\mu$:

$$\mu = \left[ 1 - \frac{\delta t}{\tau_P} ( P_0 - P ) \right]^{\frac{1}{3}} \tag{4.4.2}$$

where $\delta t$ = MD time step
$P$ = instantaneous pressure
$P_0$ = pressure of the bath
$\tau_P$ = pressure relaxation time

As with the constant temperature algorithm, the values of $P_0$ and $\tau_P$ are defined by the user (see Sections 6 and 7). It is possible to scale the pressure isotropically ($\mu x = \mu y = \mu z$) or anisotropically within SPASMS but the anisotropic method is generally recommended. Scaling anisotropically allows the box dimensions to oscillate independently of each other. A note of consequence is that the exact nature of the system in this ensemble is not known exactly. “We have not been able to prove the exact nature of the ensemble generated by this coupling method...” [Berendsen, H.J.C.; Postma, J.P.M.; Gunsteren, W.F.; DiNola, A.; Haak, J.R. (**1984**) *Journal of Chemical Physics*, *81*, 3684]. In other words, it is a reasonable assumption that this approximates a constant pressure ensemble, but it can’t be proven. Other pressure coupling schemes exist, and will be implemented in SPASMS in the future. Use of the temperature and pressure coupling would then provide an NPT simulation.

## 4.5 RATTLE CONSTRAINT

Previously we mentioned that the MD time step was necessarily many times smaller than the highest frequency motions in a molecule. The highest frequency motions are often bond stretching vibrations. Furthermore, bonds involving hydrogen generally have the greatest vibrational frequencies. Ryckaert *et al.* and van Gunsteren and Berendsen [Ryckaert, J.P.; Ciccotti, G.; Berendsen, H.J.C. (**1977**) *Journal of Computational Physics*, *23*, 327. van Gunsteren, W.F.; Berendsen, H.J.C. (**1977**) *Molecular Physics*, *34*, 1311.] have shown that it is possible to constrain the motion along bonds which allows an increase in the MD time step. Their SHAKE constraint method is utilized in the AMBER MD module. The same general procedure is derived for the velocity Verlet version of integration; this method is referred to as RATTLE. [Andersen, H. C. (**1983**) *J. Computational Physics*, *52*, 24-34.] and is used in SPASMS.

In SHAKE, the atomic positions are reset by an iterative method after each time step such that the constraint of fixed bond lengths is met to within some user defined tolerance (typically between 1-5 x $10^{-4}$ Å). In the RATTLE enhancement, the velocities are also constrained so that the velocities of a pair of bonded atoms lie entirely perpendicular to the bond between those two atoms; i.e., there is no component of velocity along (parallel to) the bond. This choice of constraints removes all motion of the bonded atoms along the bonds (i.e., bond vibrations). The use of the

RATTLE constraint has little effect on the molecular dynamical properties of molecules [van Gunsteren, W.F.; Karplus, M. (**1982**) *Macromolecules*, *15*, 1528] but it increases the efficiency of molecular dynamics by several-fold.

Suichi Miyamoto has developed and implemented SETTLE, an analytic version of SHAKE/RATTLE for TIP3P/TIP4P waters only. [Miyamoto, S.; Kollman, P. A. (**1991**) *J. Phys. Chem.*, submitted]

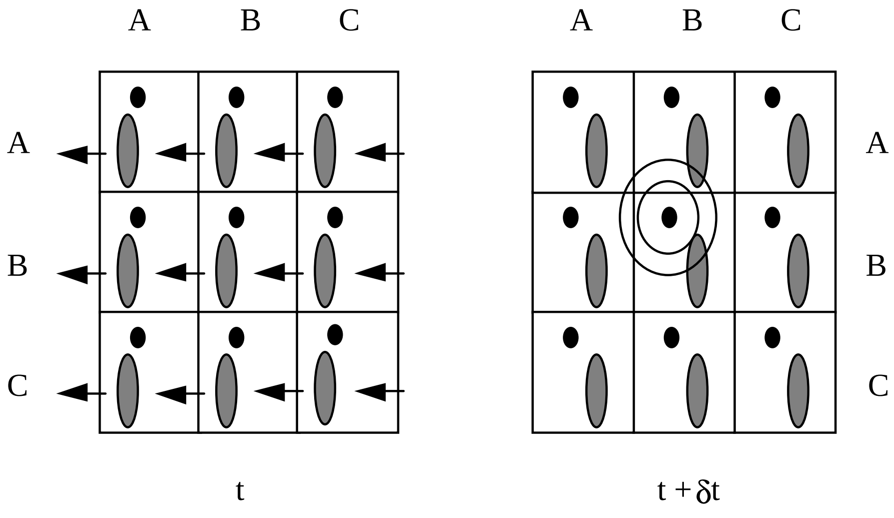
## 4.6 PERIODIC BOUNDARY CONDITIONS

In simulations of a molecule surrounded by a shell of solvent *in vacuo*, undesirable edge effects are observed at the solvent-“vacuum” interface. Periodic boundary conditions (PBCs) can be employed to eliminate these edge effects. In this technique, the ensemble of atoms is placed in a box and virtual images of the original box are constructed such that the original system is surrounded by images of itself. The atoms farthest from the center of the box would no longer suffer from interactions with an edge environment since the box now would appear to be infinitely periodic to any atom and there would be no surfaces in the system.

A number of different box types have been employed in molecular dynamics. These include the rectangular box (and its subset, the cubic box), the monoclinic box, and the truncated octahedron. At present SPASMS supports only the rectangular box for PBCs.

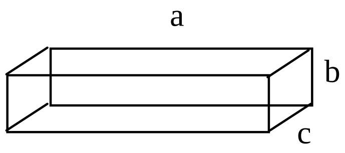
Since the periodic images are exact copies of the original box, movements of atoms in the box are duplicated in the images. The following drawing (Figure 4.6.1) represents two snapshots of the molecular dynamics trajectory of particles in a two-dimensional system with PBCs. The actual box is depicted as cell {B,B} and its periodic images are the other eight boxes. As one can see, when a particle’s movement causes it to leave a box, it is replaced by its periodic image entering from the opposite box side. Thus, the number of particles in the box is conserved.

(4.6.1)



The nonbonded cutoff needs to be chosen carefully when employing PBCs. It is essential that the cutoff be small enough that a given particle cannot interact with another particle *and* the second particle’s virtual image simultaneously. For a rectangular box, the following equation (Equation 4.6.2) can be used to calculate the nonbonded cutoff if one is using the atom based pair-list:

(4.6.2)



$$\text{cut} < \frac{1}{2}\left( x - VDW_{\text{solvent}} \right)$$

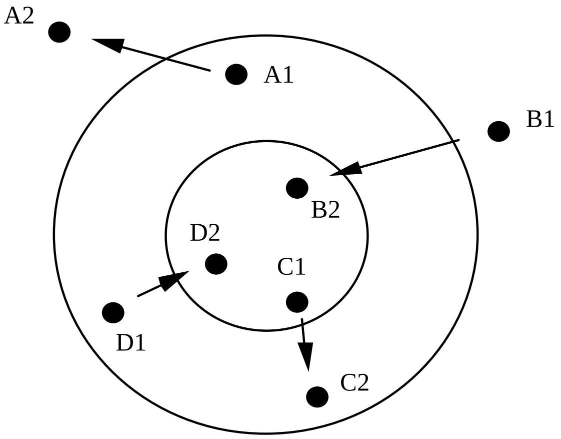
where cut = nonbonded cutoff
x = smallest box dimension (in this case, dimension *b* )
$VDW_{\text{solvent}}$ = van der Waals radius of the solvent

One should be aware that when the nonbonded pair list is stored as atom-atom pairs (NTN = 3, or atom for the nonbonded cutoff choice), rather than as residue-residue pairs, it is always possible for parts of a residue to image differently to different atoms in another residue. On the surface, this does not seem unreasonable. However, in systems that have charged atoms, one might accidentally “split” a dipolar group. This would cause some atoms to “feel” an unbalanced positive charge and other groups to feel an unbalanced negative charge. The resulting forces, even at 8 or 9Å, are enormous and create a system in a non-equilibrium state. This rather severe problem can be avoided with the use of the residue based nonbonded pair list where the cutoff distance can be no more than 1/2 • [(smallest dimension of the box) - (VDW radius of the solvent) - (largest dimension of the solvent)]. This is a major advantage of the residue based pair list over the atom based pair list.

In the Figure 4.6.1, the black particle in cell {B,B} is shown with two proposed nonbonded cutoffs. The outer cutoff is more than twice the smallest box dimension and one can see that the black particle would interact with both the grey rectangle and its periodic image. When the nonbonded cutoff is reduced to an appropriate radius (the inner circle), the black particle can only interact with one rectangle – in this case, the rectangle also in cell {B,B}.

A novel feature of SPASMS is the inclusion of a buffer region in the residue-based pair list. The nonbonded pair list update can be computationally expensive. One manner by which to limit the expense is to hold a larger list (typically generated using a 2 to 6Å larger radius) and update the smaller inner list from the larger list. This is much less expensive, as only those particles from the larger list need to be included in the update search. It is important to note, however, that the cutoff for the BUFFER list should not exceed 1/2 the smallest box dimension. A schematic of the buffer region concept is in Figure 4.6.3 below. Given the nonbonded cutoff regions shown, the following can be said about a buffer region update: Assuming particle A moves from position $A_1$ to $A_2$ between non-bond pair list updates, the residue is not included in the new nonbonded pairlist, nor is it eliminated from the buffer list. Particle B is NOT included in the buffer region, so if it moves from position $B_1$ to $B_2$ during this time, the particle is not seen. Obviously, this is undesirable and the user must take caution to not allow this to happen. If particle C moves from position $C_1$ to $C_2$, it is removed from the inner pairlist. If particle D moves from position D1 to D2, the residue list is updated to include D since it has crossed into the interaction sphere.

(4.6.3)



## 4.7 CUTOFF METHODS

Because of limited computer resources and because systems with PBCs require it, we normally truncate the nonbonded potential energy of a system at some finite distance. The energy at the cutoff distance is very likely to be non-negligible. In the past, the reason to couple to a temperature bath during dynamics was to correct for energy gained or lost due to the truncated potential. Dynamics with a truncated potential produces anomalous and non-physical behavior. One symptom is that the solute (typically a protein) cools and the solvent (typically water) heats rapidly. The temperatures of the components of the system then slowly converge until the system appears to be in equilibrium, when in fact it is not.

A relatively simple method for alleviating some of the non-physical behaviors caused by imposing a cutoff is the use of a potential switching function (see Equations 4.2.4 - 4.2.6). Such functions allow the nonbonded potential energy to be turned off smoothly and systematically and thus remove the pathological effects of using a truncated potential. With an appropriately chosen switch function, the potential function is unaffected except in the region of the switch. In addition, to allow for consistent dynamics the function must be continuous at all points, and be thrice differentiable. Shown below is the nonbonded energy curve for two water molecules with the dipoles pointing in the same direction (i.e. the hydrogens on both waters point in the same direction from the oxygen). The potential at the cutoff is relatively small ( < 0.5 kcal/mol); however, this is the greatest of error in the simulation. Also shown below is the switching function, and the "switched" potential.

The user is required to specify the radius to begin switching the potential, the radius at which the potential should be switched to zero, and the cutoff distance for the nonbonded pair list. It is possible and HIGHLY RECOMMENDED that the cutoff distance be 1-2 Å larger than the upper limit of the switch. In effect, this creates a second buffer region, distinct from the one presented above, in which all interactions are calculated as zero. This means that slightly greater simulation time is required, but allows less frequent updates of the pair list. Initial empirical studies show that for a 0.001 ps time step size, updates of the buffer region every 40 time steps yields a simulation that conserves energy.

## 4.8 ________________ INITIALIZATION, EQUILIBRATION, AND DATA COLLECTION

A MD calculation is generally initiated with coordinates refined by minimization. Velocities corresponding to a Maxwellian distribution at approximately 10-50 K are randomly assigned to each particle in the system at the start of the MD run. In a constant pressure simulation, the system warms to the target temperature according to coupling methods in Section 4.4.

The first part of the MD trajectory is called the equilibration period. During this time, the properties of the system are not constant. It may take on the order of 100 psec for the system to approach equilibrium [Levitt, M.; Sharon, R. *PNAS* (**1989**) *85(20)*: 7557-7561], but other thermodynamic properties are normally fairly stable after 10-20 psec. The energies, coordinates, and velocities which are collected during the equilibration period are not used for subsequent analyses of the system properties.

How does one determine when the system is in equilibrium? If the potential energy is plotted with respect to time during a constant temperature run, then the end of the equilibration period is found by determining at what point the average potential energy becomes constant. Similarly, during a constant pressure MD simulation, the pressure will oscillate about some point after equilibrium is reached. Other properties of the system will also become constant after equilibration unless a major conformational change occurs in the system under study. The MD output “MDEN” file contains the potential energy and pressure data, along with numerous other quantities of interest. It is very important that the user write this file every 1-2 steps during the MD simulation and plot the energy quantities to determine the equilibration period.

```
Title = " Crambin in TIP3P water using RATTLE "
timelimit =         999990.000          ! maximum cpu seconds
steps =     100                         ! number of steps per run
dt =    0.001                           ! time step size (ps)
prtfreq =     500    ! freq to write to output
edumpfreq = 500                         ! freq to write out
energies
nbupdate =     20    ! freq to updating the NB pair list
pbc = volume                            ! constant volume
periodic boundary
watermodel = tip3p   ! use fast tip3p routine
coordinateinput = crd                   ! initial coordinates
read from inpcrd
initvelocities = random                 ! randomly assign startup
velocities
tempi =  300.00000   ! initial temperature
tcontrol = none                         ! no temperature control
bufferupdate =      9999                ! freq to update buffer
region
pairlist = residue   ! use a residue based pair list
cutoff =    8.00000 ! nonbond cutoff distance
rattle = hydrogen    ! rattle all bonds with hydrogen
dielectric =       1.00000              ! dielectric constant
firstwater =       47                   ! pointer to 1st water
rbondtolerance =    0.0005              ! rattle bond length
tolerance
rvelocitytolerance =    0.0005          ! tolerance for velocity
along bonds
```

# SECTION 5: FREE ENERGY PERTURBATION

## 5.1 OVERVIEW

Free energy perturbation is a relatively new technique in molecular dynamics or Monte-Carlo simulations. Using this method, one can determine the relative Gibbs or Helmholtz free energy differences between two species. To do so, the potential energy function of one species is slowly changed or perturbed to that of a second species through a perturbation parameter λ. The free energy change will be equal to the sum of the energy differences during each step of the perturbation. The free energy will be of a Gibbs type if the isothermal-isobaric ensemble (NPT) is used and the difference will be a Helmholtz free energy if one uses a canonical ensemble (NVT).

In Figure 5.1.1

(5.1.1)

$$\begin{tikzcd}
Cl^-_{(g)} \arrow[r, "\Delta G_1"] \arrow[d, "\Delta G_3"'] & Br^-_{(g)} \arrow[d, "\Delta G_4"] \\
Cl^-_{(aq)} \arrow[r, "\Delta G_2"'] & Br^-_{(aq)}
\end{tikzcd}$$

the reactions 1 – 4 form a closed thermodynamic cycle. Since *G* is a state function, the calculation of the relative solvation free energy of $Cl^-$ and $Br^-$ is independent of the path chosen (Equation 5.1.2).

(5.1.2)

$$\Delta\Delta G = \Delta G_2 - \Delta G_1 = \Delta G_4 - \Delta G_3$$

Reactions 3 and 4 would be difficult to simulate. However, the relative solvation free energy can rather easily be calculated as the difference between reactions 2 and 1. The perturbed state at any point in reaction 1 can be represented by the following (Equation 5.1.3) potential:

(5.1.3)

$$V_\lambda = \lambda V_{Br^-} + \left(1 - \lambda\right) V_{Cl^-}$$

Figure 5.1.4 illustrates the thermodynamic cycle for the free energy difference in binding of two ligands to a protein. In this case, reactions 1 and 2 could be approached experimentally but a theoretical simulation would be difficult. This is because the ligand would have to displace solvent molecules or intramolecular protein interactions in order to bind. This displacement would require extremely long trajectories for sampling. However, the reactions 3 and 4 could be used to calculate the binding difference.

(5.1.4)

$$\begin{tikzcd}
\text{ligand}_1 + \text{protein} \arrow[r, "\Delta G_1"] \arrow[d, "\Delta G_3"'] & \text{ligand}_1 : \text{protei} \arrow[d, "\Delta G_4"] \\
\text{ligand}_2 + \text{protein} \arrow[r, "\Delta G_2"'] & \text{ligand}_2 : \text{protei}
\end{tikzcd}$$

## 5.2 PERTURBATION ALGORITHMS

SPASMS uses thermodynamic perturbation to calculate the relative free energy differences between two species. As mentioned above, the potential ($V(\mathbf{q})$) or Hamiltonian ($H(\mathbf{p},\mathbf{q})$) of the first state is coupled to the second state through the parameter $\lambda$. In initial state of the system, $\lambda$ equals one and $\lambda$ is equal to zero in the final state. The Gibbs free energy is also a function of $\lambda$ (Equation 5.2.1).

$$G(\lambda) = -kT \ln \Delta(\lambda) \tag{5.2.1}$$

where k = Boltzmann's constant
T = temperature
$\Delta$ = isobaric partition function

The Gibbs free energy between two species is then (Equation 5.2.2):

$$\Delta G_{BA} = G(\lambda_B) - G(\lambda_A) = -kT \ln \left\{ \frac{\Delta(\lambda_B)}{\Delta(\lambda_A)} \right\} \tag{5.2.2}$$

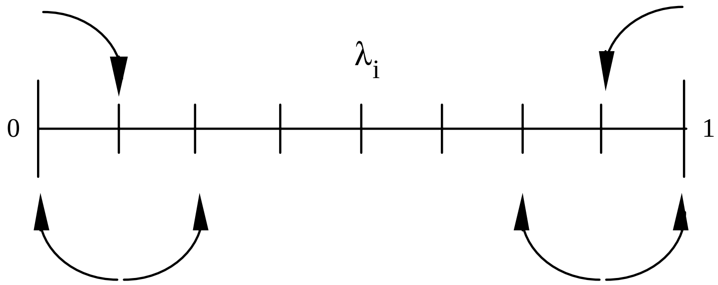
This can also be written as the ensemble average (Equation 5.2.3):

$$\Delta G_{BA} = -kT \ln \left\{ \left< \exp\left[ -\left( H\left(\mathbf{p}, \mathbf{q}\ \lambda_B\right) - H\left(\mathbf{p}, \mathbf{q}\ \lambda_A\right) \right) / kT \right] \right>_{\lambda_A} \right\} \tag{5.2.3}$$

## 5.3 DOUBLE WIDE SAMPLING

The free energy difference between two states is calculated by incrementing $\lambda$ through a series of small steps or windows. In order to sample adequately, the free energy difference between two adjacent windows should be less than 2kT. The system is equilibrated at each window and then data is collected. The total free energy change, during the simulation, is equal to the sum of free energy differences between the initial and final states of each window.

(5.3.1)



SPASMS uses a double wide sampling approach to windowing (Figure 5.3.1). The windows are sampled in the backward and forward direction at each $\lambda_i$. In addition, there is overlap sampling as depicted in Figure 5.3.1. Thus, there will be (n + 1) windows calculated during each simulation of (n) windows.

The reliability of free energy methods is well known to heavily depend on the sampling. While most would agree that longer simulation/sampling times correspond to more reliable estimates, it is not always possible to obtain converged, or even near converged averages in equation 5.2.3. To address this problem, we have implemented a free energy algorithm based on Acceptance Ratio methods to optimize the estimate. This approach was first described by Bennett (J. Comput. Phys. **22** (1976) 245) and is discussed and compared with other methods in further detail by Ferguson (J. Chem. Phys. **99** (1993) 10086) ). Essentially, the scheme optimizes the overlap between samples from adjacent windows during the simulation by reducing the variance in the estimate. The basic AR method represents the free energy change as a ratio of ensemble averages where an arbitrary weighting function has been introduced into the configurational integrals

$$\frac{\Delta_A}{\Delta_B} = \frac{\left\langle W\, exp\,(-\frac{H_A}{kT})\right\rangle_B}{\left\langle W\, exp\,(-\frac{H_B}{kT})\right\rangle_A} \quad . \tag{1}$$

By choosing $W$ in such a way as to minimize the variance in the partition function ratio, the follwoing sampling equation can be derived

$$\frac{\Delta_A}{\Delta_B} = \frac{\left\langle F(C - \frac{1}{kT}(H_B - H_A))\right\rangle_B}{\left\langle F(\frac{1}{kT}(H_B - H_A) - C)\right\rangle_A} exp(\, C\, ) \tag{2}$$

with the requirement that

$$C = ln\,\frac{\Delta_A}{\Delta_B} \quad . \tag{3}$$

where $C$ is a shift constant to the configurational energy and $F$ is the Fermi function. These simultaneous (eqs. 2 and 3) equations are solved iteratively until $C$ converges, minimizing the variance and optimizing the estimate of the free energy for each window in the simulation.

These equations have been implemented in SPASMS as part of the double-ended sampling protocol of statistical mechanical perturbation theory. Although the applications have been limited, Ferguson has shown that better estimates are possible using this algorithm given limited sample sizes and/or large molecular changes. In addition, it has also been shown that the methodology can yield information regarding the shape and overlap of the energy samples at each window, allowing the window sizes to be optimized for specific systems amd molecular changes.

```
Title = "Perturbation Test Case Neon to Argon in 216 TIP3P H2O"
timelimit =      999000.000             ! maximum cpu seconds
lambdai = 0.0
deltalambda = 0.125
runs =      2                           ! number of runs
steps =     500                         ! number of steps per run
fepequil =  500
dt =    0.00050                         ! time step size (ps)
coordinateinput = ( crd ,  box )        ! init coords from inpcrd
prtfreq =     250                       ! freq /write output
qdumpfreq =    2000                     ! freq /write coord file
edumpfreq =    2000                     ! freq /write energy file
nbupdate =     10                       ! freq /update NB pairlist
bufferupdate =  99999                   ! freq /update buffer region
dielectric =    1.00000                 ! dielectric constant
rmcomminterval = 99999                  ! freq /rmv ctr/mass motion
!
! Temperature control
!
tempi =  300.00000                      ! initial temperature
tcontrol = random                       !
velreassign =     500                   ! freq to randomly reassign vel
minorcollfreq =    0.00000              ! freq for minor collisions
!
! Pressure
!
pbc = pressure                          ! constant pressure
pcontrol =   anisotropic                !
taup = 8.00000                          ! pressure coupling
comp = 44.6                             ! water compressibility
!
! Water model, pairlist type and switch
!
pairlist = residue                      ! use a residue based pair list
firstwaterresidue     =      2          ! for fast water routines
watermodel = tip3p                      ! use tip3p water
cutoff =    8.00000                     ! nonbond cutoff distance
switchoption ( 7.00 , 7.50 )            ! switch off nonbon from 7-7.5 A
!
! Rattle
!
rattle = all                            ! rattle all bonds
rbondtolerance =    0.000001            ! rattle bond length tolerance
rvelocitytolerance =    0.000001        !
```

## 6.1 OVERVIEW

The SPASMS control input is read from the MDIN file. The input is based on a keyword driven parser. As such, it can interpret input files which are not constrained to any specific format. Rather, these files must conform to a given syntax. The syntax is quite consistent and straightforward: the basic rule is that arguments always follow their associated keywords. The exception to this rule is for keywords which are classified "logical" keywords; these do not take *any* arguments. For keywords which can have multiple arguments, the argument list is enclosed in any combination of the following delimiter characters: parentheses [ ( ) ], single [ ' ' ] or double [ " " ] quotes, or forward [ / ] slashes. Arguments may be separated from their keywords by one or more spaces, equal [ = ] signs, commas [ , ], or semi-colons [ ; ]. There must be at least one such delimiter between the keyword and its argument. There can be as many keywords and arguments as fit on a line and keywords need only be specified by enough characters to uniquely identify them. In the SPASMS documentation, the uniquely identifying characters are written in upper case. The parser itself is not case sensitive so any combination of upper and lower case letters will be accepted for keywords. If there is no occurrence of a keyword in the MDIN file, the established default value for that option remains. Comments are indicated by exclamation [ ! ] points and these may be placed anywhere. When an exclamation point is encountered, the remainder of the line is considered to be a comment.

This documentation section is arranged such that the individual keyword options are first listed, in alphabetical order, and discussed. A discussion of *group input* then follows. Finally, a flowchart for choosing keyword options is given. The keywords are described in detail and their argument types and default arguments are given. Background information is also given in order to provide the user with some knowledge as to when to use the options and also so keyword arguments can be chosen intelligently. The keywords fall into four classes: those that take floating point arguments, those that take integer arguments, those that effect boolean ("logical") values, and those that take complex arguments or set many options. In the following discussion, the keyword type is indicated in the default argument list by the letters 'f' (f̲loating point), 'i' (i̲nteger), 'b' (b̲oolean), or 'm' (sets m̲any options).

## 6.2 KEYWORD OPTIONS

BELly (b; FALSE)

A user may specify that during a simulation only certain atoms are allowed to move. These are the set of belly atoms. If this option is set, then a *group option input* that describes the set of belly atoms will follow the *endofinput* keyword. The user should beware when using the residue-based pair lists that the belly does not constrain only one atom of a residue, leaving the others free to move. Belly atoms are specified in formatted group input at the end of the input file.

BETa (f; 90.0)

*Beta* is the angle between the (non-)orthogonal walls of a PBC box. In SPASMS, *beta* should be set to 90.0 as the option for noncubic boxes is not yet implemented.

BUFFERRegion (f; 0.0)

The BUFFERRegion keyword gives the radius (angstrom) of the buffer region. This option should be used only if the pair list option is *residue*. A suggested radius of the buffer region is 2 - 6 Å longer than the nonbonded cutoff, since it takes no additional time to generate the list. However, the larger the value of *bufferregion*, the larger the list that has to be searched between full updates of the pair list. This of course increases the time needed to search the list.

BUFFERUpdatefrequency (i; 1)

The number of steps between updating the buffer region is established with the keyword *bufferupdatefrequency*. This only applies if the pair list is residue based (which includes the *residue* and *water* arguments of the *pairlist* keyword). The buffer region update frequency should be fairly high as the list in this region is not large and one may want to guarantee energy conservation (e.g., when using switches).

CAPAtompointer (i; 0)

If the *capoption* is *modify*, then this keyword overrides the cap atom pointer (the index of the first cap atom) from the PRMTOP topology file. Otherwise, this keyword is ignored.

CAPForceconstant (f; 0.5)

This keyword sets the cap constraint force constant. Typical values range from 0.3 to 100.0 kcal/mol•Å$^2$. It is recommended that *smallest* possible values be used while preventing the solvent atoms from drifting appreciably away from the solute.

CAPOption (m; INACTIVE)

A cap is a spherical group of solvent – usually water – molecules. The cap is created using the AMBER EDIT module or the LEaP program. Restraining forces are applied to the cap atoms so the solvent will not drift appreciably away from the solute during the simulation. Unlike *restraints* applied to solute positions, no *group input* has to be given for the *capoption*. Options:

| | |
|---|---|
| *active* – | The cap will be in effect if specified in the PRMTOP file. |
| *modify* – | The cap will be in effect but the cap atom index of the PRMTOP file will be modified with the *capatompointer* keyword. |
| *inactive* – | Regardless of the value of the cap option passed in the topology file, the cap will not be in effect. |

CODrop (f; 0.0)

For shifted potentials, this keyword sets the adjusted energy at the cutoff. The keyword is intended primarily as a debugging tool.

COMpressibility (f; 44.6)

This keyword indicates the inverse of the compressibility of the system ($10^{-6}$Bar$^{-1}$) for constant pressure simulations. If the compressibility of the material under study is not known, or the simulation is to be performed in water, a value of 44.6 is recommended.

CONvergencecriteria (f; 0.1)

During minimization, if the norm of the energy gradient is less than *convergencecriteria*, the calculation will terminate after recalculating the nonbonded pair list and all of the energy components.

COOrdinateinput (m; CRD)

This keyword specifies the information to be read from the input coordinate [INPCRD] file. The coordinates are stored first and this must be specified with the keyword. The velocities, if any, are stored next. If a simulation has been run with reference constraints, those positions follow the velocities. Finally, in the case of a simulation with periodic boundary conditions, the box size is stored. In all cases except the coordinates, the items do not appear in the INPCRD file until after the simulation has been completed. There are several choices of options:

| | |
|---|---|
| *crd* – | The initial Cartesian coordinates are read from INPCRD. |
| *vel* – | The initial velocities are read from INPCRD. |
| *box* – | The initial box dimensions are read from INPCRD. |
| *refc* – | The reference Cartesian coordinates are read from INPCRD. |

CUtoff (f; 8.0)

The nonbonded cutoff radius (angstrom) is set by this option. For very small molecules calculated *in vacuo*, an infinite cutoff (99 Å) is typically used while the 8 Å default is often used for larger systems. However, if one is using switching functions, it is recommended that the cutoff be about 1 to 2 Å longer than the upper cutoff distance for the switch so that the pair list does not have to be regenerated every step.

DDd (b; TRUE)

This keyword sets the type of dielectric function to be used in a simulation. If the keyword is set, a distance dependent dielectric will be used in the calculation. One usually sets the distance dependent dielectric in order to simulate the effects of an implicit solvent. The constant dielectric function is usually appropriate for calculations which include explicit waters or during which one would like to simulate a vacuum.

DELTALAMBDA (f; 0.0)

The window size for free energy perturbation.

DIElectricconstant (f; 2.0)

This is the dielectric multiplicative constant for electrostatic interactions and it is coupled to the type of dielectric function, *ddd*. As an example, to obtain a dielectric constant of $4r_{ij}$, the *dielectricconstant* would be equal to '4' and the *ddd* keyword would be set.

DIFfrescutoff (b; FALSE)

It may be desirable to have different nonbonded cutoff distances for different residues. If one chooses to apply different cutoffs, then the *diffrescutoff* keyword should be set. Following any belly option *group input*, the residues having different nonbonded cutoff distances are read with a cutoff and with a lower switch value (if switches are used).

DDRIVE (b; false)
DIHEDRALDRIVER (b; false)

synonymous keywords — implies the dihedral driver option is to be used. Requires further input from input file MDDRV. This file specifies the dihedralsto rotate and other control variables. The first line specifies a title for the input deck. The second line defines the control variables and uses the format:

```
read(iodrv,'(i10,2i5,2f10.4)')ndrv,idrout,idrmin,edrcut,ddrvpk
```

ndrv defines the number of dihedrals to read in and rotate through and may be a mximum of 15.

idrout controls output style for listing: 0 for short listing, 1 for long.

idrmin provides a mechanism to perform the search with NO minimization. A value of 0 will produce optimized conformers where 1 means NO optimization will be performed. The latter can be used to enumerate structures rapidly.

edrcut specifies the energy cutoff for conformer storage. If the energy of the conformer is above edrcut, the structure is not saved, the search continues.

ddrvpk specifies the constraint peak height. A value of 100 kcal/mol is used if this value is negative. This option would allow the user to make a crude conformational search when combined with edrcut.

The third line (3a.......) inputs the dihedrals ndrv times. The format is (5i5,2f10.4) where the first four integers specify the dihedral i,j,k,l, the fifth integer specifies the number of increments, of rotations to perform. The first real number defines the initial configuration to start the dihedral and the second real number defines the dihedral increment. (A value of 999. for the former indicates to start the search at the current configuration.)

```
Dihedral Driver Input Sample for Heptane
         4    1    1  -0.1     -9.0
    1    2    3    4    2   180.0      120.0
    2    3    4    5    2   180.0      120.0
    3    4    5    6    2   180.0      120.0
    4    5    6    7    2   180.0      120.0
```

The above input file will configure heptane in the all trans arrangement and rotate through the gauche- and gauche+ conformers resulting in 3**4 configurations. No minimization.

**Note**: In SPASMS, the dihedral angle associated with atoms i, j, k, and l is determined by sighting along the j-k bond FROM atom j TO atom k. Imagine two vectors, the first points from atom j to atom i (rij), and the second points from atom k to atom l (rlk). In this position, if the rij vector overlaps (is parallel to) the rlk vector, the dihedral angle is zero. If the vectors are anti-parallel, the dihedral angle is 180 degrees. If rij must be rotated (about rkj) clockwise to overlap rlk, the dihedral angle is positive.

DIHEDRALDUMP (b; FALSE)

if used, this allows the user to dump a series of files from a dihedral driver run. At the end of each minimization, a file (in standard format unless USEPDBDRIVEDUMP is set) is created and the current coordinates are saved.

DRIVEEXTENSION (c; DRV)

The file extension to be used if dumping coordinates at the end of each minimization with the dihedral driver option.

DT (f; 0.001)

This parameter indicates the time step size (psec) for the simulation. Empirical tests indicate that values of 0.0010 – 0.0015 are reasonably safe when the RATTLE options are selected, however, values as small as 0.0001 psec may be needed. If RATTLE is not used, the typical range for the time step is 0.0005 – 0.0010. Note that for energy conservation, one might have to use a step size substantially smaller than the recommended values.

ENdofinput (b; FALSE)

This keyword is used to terminate reading the keyword input. To specify *group input*, such as belly or constraint groups, terminate the keyword input with the *endofinput* keyword and follow it with group specifications.

ENSEMbledynamics (b; FALSE)

Turns on the ensemble dynamics option. Requires input in group format following the ENDofinput line.

FEPEQUILibrate (i; 0)

Equilibration steps for free energy perturbation.

Firstwaterresidue (i; 0)

This keyword is used only if the special water potential routines (*watermodel*) are to be used. *Firstwaterresidue* is the index of the first water residue.

FLATwell (b; FALSE)

Turn on the NMR processing code. Requires further input from file MDNMR. The reference for the techniques in this implementation is W. C. Swope and D. M. Ferguson, J. Computational Chemistry, 1992. This feature is used for simulations related to NMR studies. It may be used along with dynamics, minimization, or free energy simulations. The feature serves to constrain particular inter-atomic distances and/or dihedral angles to a predetermined range. It is invoked by entering the keyword FLATWELL in the input file (named MDIN in the shellscript). This keyword may be in any case and may also be shortened up to the point that it is still unique among all the other keywords in use. For example, "flat" would also work. If the keyword is found in the input file, a second file (named MDNMR in the shellscript) is searched for the parameters that control this aspect of the simulation. For example,

```
-------------this line is not part of the MDNMR file------------
NMR Input - to be used with butane parameter and coordinate files
        1         1
    0    0    1   11   3.8       4.1      25.0      25.0       1.0
    1    5    8   11 170.0     190.0      25.0      25.0       1.0
-------------this line is not part of the MDNMR file------------
```

The first line is a title that is copied into the output file for the run. The second line specifies how many flatwell constraints are to be specified of each type, first the bond flatwell energy terms, then the dihedral angle flatwell energy terms. The example file above has one constraint of each type. The bond constraint is between atoms 1 and 11; the dihedral constraint is associated with atoms 1, 5, 8 and 11. The next values on each line represent the minimum and maximum values that can be taken by the coordinate under consideration. In the file above the distance between atoms 1 and 11 is to be held to a value between 3.8 and 4.1 ANGSTROMS. Similarly, the dihedral angle associated with atoms 1, 5, 8, and 11 is to be held to values between 170 and 190 DEGREES.

The next two values on each line specify the spring constants associated with the restoring force that is applied when the atom-atom distance or dihedral angle goes outside the specified range. The first number pertains to the lower bound, and the second pertains to the upper bound.

The last number on each line is a weighting factor that is included for future capability but is not used now.

The potential applied by the "bond" term is if r(ij) is less than rmin,
U = 0.5 * kmin * (r(ij) - rmin)**2 if r(ij) is greater than rmax,
U = 0.5 * kmax * (r(ij) - rmax)**2 and U = 0 otherwise.

The potential applied by the dihedral angle term is
if phi(ijkl) is less than phi(min), U = 0.5 * kmin * ( 1 - cos(phi(ijkl) - phi(min)) )
if phi(ijkl) is greater than phi(max), U = 0.5 * kmax * ( 1 - cos(phi(ijkl) - phi(max)) )
and U = 0 otherwise.

The title line is read with FORMAT(A80), the second line with FORMAT(*), and the third and subsequent lines with FORMAT(4I5,5F10.4).

Genpairlist (b; FALSE)

This flag can be used to generate a nonbond pair list. It could be used to avoid storage of the pairlist, where either an infinite cutoff is being used (no need to store the list, presumably) or if the pair list is being updated every time step. It was added for compatibility with AMBER and for future capability with the systolic force-engine, when they put the pair list generation on the force engine.

Initvelocities (m; RANDOM)

This parameter indicates the source of the initial velocities. An initial molecular dynamics or free energy perturbation run would be started with the *random* option. However, if the run needs to be restarted, the *input* option would be used so not to perturb the steady-state of the system. Options:

| | |
|---|---|
| *random* – | The initial velocities are assigned randomly from a Boltzmann distribution. |
| *input* – | The initial velocities are read from the INPCRD file |

Note: A value of *random*, however, will overwrite the velocities read in if *vel* or *box* is specified for the coordinate input. This would allow the user the option of starting an independent run with updated box dimensions.

LAMBDAInitial (f; 0.0)

In Free Energy Perturbation, this is the initial value of lambda.

MAxrattleiterations (i; 1000)

This indicates the maximum number of iterations per step that can not be exceeded in the determination of the constraint positions in RATTLE. In most cases, the constraints are met within 10 – 20 iterations.

MINImization (b; FALSE)
If a minimization is to be performed, this flag should be set. The maximum number of cycles of minimization is specified as the number of steps per run (*steps*).

MINOrcollfrequency (f; 0.1)
*Minorcollfrequency* sets the frequency of stochastic collisions per particle per time step for temperature control. The default value of 0.1 would result in one particle collision every 10 steps. If the *RATTLE* option is used, the collisions are performed on a residue basis to conserve energy.

MONTecarlo
not used in this version.

MULTIPLEIN (b; FALSE)
Use multiple input files with one topology (parm) file. This implies the MULTISTART and MULTILAST keywords are to be used.

MULTIPLEOUT (b; FALSE)
Use multiple output files with one topology (parm) file. This implies the MULTISTART and MULTILAST keywords are to be used.

MULTISTART (i, 0)
The first number appended after the rootname and before the extension in creating file names for multiple input and output. On Unix machines, a file is created with the name root<i>.ext where i is between multistart and multilast.

MULTILAST (i; 0)
The last file number. (See Multistart for details)

NBupdatefrequency (i; 1)
*Nbupdatefrequency* indicates the number of steps between updates of the pair list. The desired value is 1. However, realistic values would be less than 20 for MD and FEP simulations without switching functions and less than 100 for minimization. In simulations with switching functions, one should choose *nbupdatefrequency* so that energy is conserved.

NDfmin (i; 6)
This parameter specifies the number of degrees of freedom that will be subtracted from the total number of degrees of freedom in the system. If the center of mass motion is to be removed, then this option might be set equal to six. If not, then the parameter should be set to zero. This parameter should also be zero during belly dynamics as the program will automatically calculate *ndfmin*. In general, if one does not have to remove six degrees of freedom, then one will have a simulation which is 1/2 kT higher in energy for each degree of freedom different from six. Since the interesting value is usually the ensemble average, this excess energy will simply raise the ensemble average. The equilibration run will be the same, and the collection run will be the same. (see *dontremovecomm*)

NPT (m)

One can specify a molecular dynamics or free energy perturbation run using the NPT ensemble with this keyword. The temperature control is achieved using stochastic collisions.

NVE (m)

The *NVE* keyword can be used to specify a molecular dynamics or free energy perturbation run using the micro-canonical ensemble; there is no temperature control for this ensemble.

NVT (m)

One can specify a molecular dynamics or free energy perturbation run using the canonical ensemble with this keyword. The temperature control is achieved through the use of stochastic collisions during the simulation.

OMITBondinteractions (b; FALSE)
OMITHinteractions (b; FALSE)

In simulations with RATTLE constraints, one should eliminate from the calculation those bonds that are being constrained. The *omitbondinteractions* flag is set in order to ignore all bond interactions while the *omithinteractions* parameter indicates that all bonds to hydrogen are omitted from the force calculation.

PAirlist (m; RESIDUE)

Several pair list options exist. *All* pair lists are *generated* from residue-residue interactions. That is, given two residues *R* and *S*, if *any* atoms *i* (in *R*) and *j* (in *S*) are within the cutoff distance, then the two residues are said to interact. The differences in the pair list options lie in the manner in which the pair list is *evaluated*.

| | |
|---|---|
| *atom* - | Given two residues *R* and *S*, if any atoms *i* (in *R*) and *j* (in *S*) are within the cutoff distance, then the two residues are said to interact. For all atoms *i* in *R* and all atoms *j* in *S*, the pair *i-j* is stored. The pairs of atoms are stored in a large, memory inefficient pair list.<br>Once the pairs of atoms are stored, they lose all connection with the remaining atoms in the residue. This prohibits the use of correctly implemented residue switching functions, among other things. It is also much easier to split dipoles while using the *atom* option. However, the *atom* option is sometimes faster than the *residue* option. In particular, for small systems (<1000 atoms) with infinite cutoffs, the *atom* routine ('nb1st') runs 10-15% faster than the *residue* routines ('nb1nex' and 'nbexcl'). |
| *residue* - | Current implementation: Given two residues *R* and *S*, if any atoms *i* (in *R*) and *j* (in *S*) are within the cutoff distance, then the residue pair *R-S* is stored in a residue pair list. For water-water interactions, that means checking 9 atom pairs total. (Actually, the routine will 'break out' of the loop if it finds two pairs within the cutoff distance, so it is typically less than 9.) However, if the input option *firstwaterresidue*, then only the O-O distance is used. See description below. |

The routine ‘nbrsgn’ is used for the *residue* option. This method is much more memory efficient than the *atom* pair list. The two methods are degenerate if there is only one atom per residue but this is not usually the case.

In the following discussion, ‘protein’ implies all non-water residues. Given two protein residues *R* and *S*, if any atoms *i* (in *R*) and *j* (in *S*) are within the cutoff distance, then the residue pair *R-S* is stored in a residue pair list. For all protein-water interactions, if any atom in the protein (residue *R*) is with the cutoff distance of the oxygen of water (residue *S*), the residue pair *R-S* is included in the pair-list. For all water-water interactions, if the O-O distance is within the cutoff distance, then the residue pair *R-S* is included in the pair list. The user can override the last two alterations by *not* specifying the *firstwaterresidue* keyword argument. The system is then treated as if there was *no* water in the system and the water-water interactions are evaluated in routine ‘nbrsgn’, just like the protein-protein interactions. Thus, it is crucial that the user specify the *firstwaterresidue* keyword argument correctly *if* there is water in the system.

The pair list is generated in routine ‘nbrsgn’ and evaluated in routine ‘nb1nex’ for protein-protein and protein-water interactions. For water-water interactions, the pair list is generated in ‘wtrsgn’ and evaluated in either ‘watp3p’ or ‘wtip3p’ for *watermodel* AMBER TIP3P or Jorgensen’s TIP3P model, respectively.

| | |
|---|---|
| *firstatom*- | Current implementation: In this method, the distance between the first two atoms in a pair of residues is checked to determine if the residue pair *R-S* is to be stored. If you specified ‘firsatom' with a protein, it will check the $N_{backbone}$-$N_{backbone}$ distance and either include or not include the entire residue in the pair list based upon this distance. NOT GOOD! This method is also quite dangerous if short cutoff distances are to be used. The routine ‘wtrsgn’ is used for this option. |
| *specialatom* | As in firstatom, but checks the atoms used to determine the switches. |

PBc (m; NONE)

The type of periodic boundary condition is specified with this keyword. There are several options:

| | |
|---|---|
| *none* – | Even if the parameter file has PBC information, no periodic boundary condition is applied. |
| *volume* – | A constant volume simulation, with periodic boundary conditions, will be performed. |
| *pressure* – | A constant pressure simulation, with periodic boundary conditions, will be performed. Variables PCONtrol and TAUP must be specified if this option is used. |

PControl (m; NONE)

Pcontrol specifies the pressure control (coupling) scheme during a molecular dynamics or free energy perturbation simulation. The options are:

*none* – No pressure coupling is employed during the simulation.

*isotropic* – During the simulation, the pressure is held constant and the box sides in a periodic system are to be scaled equivalently.

*anisotropic* – During the simulation, the box sides in a periodic system are to be scaled independently to maintain pressure. This is the recommended method.

PREference (f; 1.0)

This keyword specifies the external pressure (Atmospheres) on the system during a constant pressure run.

PRTfrequency (i; 9999)

This keyword indicates the number of steps between writing to the output file MDOUT. The MDOUT file provides diagnostic information obtained during the course of a simulation.

RAttle (m; NONE)

This keyword specifies the RATTLE choice. Options:

*none* – No bonds are constrained during the simulation.

*hydrogen* – During the simulation, all bonds involving hydrogen atoms are to be constrained with RATTLE and all other bonds are to remain unconstrained.

*all* – All bonds are to be constrained to the minimum energy bond length during the simulation.

RBondtolerance (f; 0.0001)

During the RATTLE constraint iterations, the bond lengths will all be set to the equilibrium values within this tolerance (in Å). Values for this tolerance are system dependent, and are also dependent upon the amount of energy conservation one might desire. For instance, in order to have energy conservation with switching functions in water with a 0.001 psec timestep size, this value might have to be set as low as 0.00003 Å, while in a protein employing velocity scaling, this value might be as high as 0.0005 Å.

RDFOption (m)

This keyword sets the specification of collecting radial distribution function data. The rdfs at this point are meaningful only in pure liquids, such as water or argon, and only if applied to the first atoms in the residues. The RDF option specification requires three parameters – the frequency, lower, and upper bounds. If these are specified using the *rdfoption* keyword, one must use the syntax: RDFOption = (<rdflower>, rdfupper>, <rdffreq>). Alternately, *rdfoption* can be specified with no arguments and *rdflower*, *rdfupper*, and *rdffreq* can be specified using the *rdflower*, *rdfupper*, and *rdffrequency* keywords.

RDFFrequency (i; 9999)

This keyword specifies the number of steps between collecting RDF data.

RDFLower (f; 3.0)

*Rdflower* is the lower limit for computing radial distribution functions.

RDFUpper (f; 12.0)

*Rdfupper* is the upper limit for computing radial distribution functions.

REDefineimagingatoms (b; FALSE)

It is sometimes desirable to redefine the atom on which switches are applied to make it more centro-symmetric. One can do so by setting this keyword and reading in the residue imaging atoms' redefinition as *group input*.

REFtime (f; 0.0)

*Reftime* is the time, in psec, at the start of a molecular dynamics calculation. This variable is for the user's reference only.

REMOVecomm (b; FALSE)

This flag says remove the translational and rotational motion from the initial velocities. It is generally not recommended that you remove the center of mass motion. As described for the keyword *ndfmin*, the extra kinetic energy will average out over a simulation to be 1/2 kT per degree of freedom. Removal of the center of mass motion tends to remove a tremendous amount of energy from the system. Doing this frequently can cause the system to cool off. However, if one is performing dynamics on a molecule in the gas phase, it is wise to remove the center of mass motion at the beginning of the simulation and then never more. The parameter should be set if a belly option is used. An indication that you are using a time step that is too large is that the center of mass motion during a simulation does not remain constant (i.e., the momentum of the system is not conserved).

RESTArt (b; FALSE)

The restart option is set to indicate the continuation of a job. It is used internally in SPASMS to help determine the items to be read from the coordinate file and the source for the starting velocities.

RESTRaints (b; FALSE)

The *restraints* parameter is used for applying coordinate constraints. The default option (no *restraints*) specifies that the system will not have constraints to reference positions. If the keyword is set, reference positions are to be read from the file REFC (see Section 1.6). Harmonic constraints and forces are placed on the positions of those atoms specified in a *group input* at the end of the MDIN file.

RLower (f; 0.0)

This limit is the distance at which the switch value starts for deviate from 1.0. At all distances less than this value, the full potential interaction is calculated. Between this distance and the upper radius (*rupper*), the energy and forces of all interactions between residues are scaled by the switch value. This is in turn determined from the distances of the two atoms that are first in the residues. Although the decision is system dependent, it is generally recommended that the distance between *rlower* and *rupper* be about 1 - 2 Å.

RMcomminterval (i; 999999)

The RMcomminterval keyword indicates the number of steps representing how often the center of mass motion is to be removed. It is generally recommended that the user *not* remove the center of mass motion (see *dontremovecomm*).

Sometimes, however, the user might want to remove the COM motion. For example, if a movie is to be made of protein dynamics, it would be easier to view without the motion. In this case, if the user runs with velocity rescaling (*velocity tcontrol*), the *rmcomminterval* will have to be rather short. The frequency of motion removal will be inversely proportional to the total mass of the system, since what is really happening is a lack of conservation of momentum during the simulation as well as a constant velocity in one direction. If, on the other hand, one uses stochastic collisions (MAJOR), one should be able to remove the center of mass motion after each collision and the molecule should remain centered.

ROOTDRIVE (c; DRIVER)

The rootname to use if the dihedral driver dump option is used.

RUNs (i; 0)

In molecular dynamics and free energy perturbation, this keyword indicates the number of runs into which the simulation will be divided. Because the restart information is written only at the end of a run, it is wise to break the simulation into several short runs (i.e., the runs should be of such a length that if the computer crashes, an unacceptable amount of computer time will not be lost). Also, the temperature can be rescaled exactly to the desired temperature at the end of a run. Using this value, and the number of steps per run, gives the user very tight control over the temperature.

RUPper (f; 0.0)

*Rupper* is the radius at which the switching function is set to zero. Beyond this distance, all interactions are defined as zero. Note that if different residues are to have different cutoffs (*diffrescutoff*), then the switches must also have different starting and ending points. In this case, the distances between turning on the switch and turning it off are independent of the radius and the lower value for the differing residues is specified in the group input for those residues. In all cases, the distance over which the switch is to be applied is constant for the simulation. If one has a group of atoms with a lower value of 7 Å and an upper value of 8 Å while the other set of residues has a lower value of 10 Å, the upper limit for the second group will be set at 11 Å. The nonbonded cutoff should be about 1-2 Å longer than *rupper* so that the pair list does not have to be regenerated every step.

RVelocitytolerance (f; 0.0001)

This tolerance is similar to *rbondtolerance*, but pertains to the tolerance of the velocity components along the bonds. Similar values as for the bond length tolerance are appropriate here.

SCEe (f; 2.0)

This is the scaling factor for 1-4 electrostatic interactions ($\mathrm{EEL}_{\mathrm{scale}}$ in Equation 2.3.1). The AMBER force field was derived with 2.0 as the scale factor while the OPLS force field was derived with 8.0 as the scale factor.

SCNb (f; 2.0)

This is the scaling factor for 1-4 VDW interactions ($\mathrm{VDW}_{\mathrm{scale}}$ in Equation 2.3.1). The AMBER force field was derived with 2.0 as the scale factor. The OPLS force field was also derived with 2.0 as the scale factor.

SEed (i; 71277)

The random number generator is used in several places in SPASMS. Primarily, it is used in molecular dynamics and free energy perturbation for the assignment of the initial velocities from a gaussian distribution about a specified temperature. The random number generator is also used for the stochastic temperature coupling. The generator must be initialized with a seed and that seed must not be a negative number. If one would like to run several trajectories from a single coordinate set, then the seed must be different for each calculation. Otherwise, the same trajectory will be generated for each run. A good method for assigning the seed is to use the date (i.e., 120491 for 12 April 1991).

SETTle (b; FALSE)

Use the analytical versions of SHAKE/RATTLE developed by Shuichi Miyamoto. This applies only to the water molecules (either TIP3P or TIP4P) in a simulation. It also turns on at minimum RATTLE=HYDROGEN.

STeps (i; 0)

In molecular dynamics, this is the total number of steps per run (see runs). The total simulation time is determined from to be equal to [(number of runs) × (number of steps per run) × (step size)]. During a minimization, *steps* is the maximum number of steps allowed. If this number of steps is exceeded, then the minimization is terminated.

SWitchoption (m)

This keyword allows the specification of the switch function. Typically, one should use only residue-based switches so one would choose *residue* or *water* for the *pairlist* option. Conservation of energy during molecular dynamics is achieved only if one performs this type of simulation. The switch function specification requires two parameters – 'rlower' and 'rupper'. If these are specified using the *switchoption* keyword, then the syntax: SWitchoption = (<rlower>,<rupper>) must be used. Alternately, *switchoption* can be specified with no arguments and *rlower* and *rupper* specified using the *rlower* and *rupper* keywords.

TAUPressure (f; 0.2)

This parameter refers to how tightly the system is to be coupled to the pressure bath during constant pressure simulations. In Equation 4.4.2, the constant $\tau_P$ is equal to *taupressure*. The units for this constant are $psec^{-1}$ and recommended values are in the range of 0.1 to 0.5 $psec^{-1}$.

TAUTemperature (f; 0.2)

*Tautemperature* is the temperature coupling constant ($psec^{-1}$) when using *velocity* temperature control. In Equation 4.4.1, the constant $\tau_T$ is equal to *tautemperature*. The larger this number, the more loosely coupled the system is to the temperature bath. Recommended values for *tautemperature* are in the range of 0.1 to 0.4 $psec^{-1}$.

TControl (m; NONE)

Several types of temperature control during molecular dynamics or free energy perturbation are available in SPASMS:

*none* – This option is used when no temperature coupling is employed. This would be used for a classical, or NVE, simulation.

| | |
|---|---|
| *scaling* – | The *scaling* option is used when temperature control is accomplished by the Berendsen coupling scheme of rescaling velocities. |
| *random* – | If the user would like temperature control to be accomplished by Andersen's method of stochastic collisions, then the *random* option should be used. |

TEmpi (f; 300.0)

The initial velocities, if not read from the input file INPCRD, are assigned from a gaussian distribution about this temperature (Kelvin).

TFluctuation (f; 10.0)

At the end of a 'run', the temperature of the system is scaled exactly to the reference temperature if the *scaling* tcontrol option is chosen for the temperature control and the system deviates from the reference temperature by more than the amount specified by *tfluctuation*. It would be possible to run a constant kinetic energy simulation by setting the number of steps per run to 1, the number of runs to the desi

red length of the simulation, and *tfluctuation* to a very low value, such as 0.0001K. Then, during each and every step, the velocities would be scaled exactly to the reference temperature.

TIMelimit (f; 99999.0)

The time limit for the job, in CPU seconds, is set by this option. This is a useful option if time limits have been placed on user computer queues. The keyword can also be used to prevent excessive loss of CPU time if a job takes longer than anticipated, such as when input is not set as intended. At the end of each timestep, the elapsed CPU time is checked against this number. If the elapsed CPU time exceeds this number, the simulation is terminated. Prior to termination, however, the system is characterized by recalculating the nonbonded pair list and all of the energy components. The coordinates are then written to the restart file.

TITle (m; 'Untitled')

This keyword sets the title of the run. The allowed option is any string less than or equal to 80 characters. If the string contains more than one word, it must be in single [ ' ' ] or double [ " " ] quotes, parentheses [ ( ) ], or forward [ / ] slashes.

TReference (f; 300.0)

If the simulation is not classical (NVE), this keyword indicates the desired reference temperature, in Kelvin, of the system. In Equation 4.4.1, *treference* is $T_0$.

USEPDBDUMP (b, FALSE)

Uses a pdb file format for the dumping coordinates. Used with multipleout.

USEPDBIN (b; FALSE)

Uses a pdb file format for the input file. If used with multiplein runs, each file opened is expected to be in pdb format. **NOTE: ONLY COORDINATES ARE EXTRACTED UPON INPUT. ATOMS MUST BE IN THE SAME ORDER AS THEY APPEAR IN THE PARAMETER TOPOLOGY FILE. NO TER CARDS ARE ALLOWED.**

USEPDBOUT (b; FALSE)

Uses a pdb file format for the restart file. If used with multipleout runs, each file created is in pdb format.

USEPDBDRIVEDUMP (b; FALSE)

Uses a pdb file format for the dihedral driver output files.

USER[1-5] (i; 0)

These keywords are currently unassigned. They are intended for user routines.

VDumpfrequency (i; 9999)
Qdumpfrequency (i; 9999)
EDumpfrequency (i; 9999)

These keywords determine the frequency that the coordinates (*qdumpfrequency*), velocities (*vdumpfrequency*), and energies (*edumpfrequency*) are written to the output files MDCRD, MDVEL, and MDEN, respectively. This output can later be used for analysis of the trajectory. Note that the dumps should be an integral of the number of steps per run.

VELReassignment (i; 9999)

*Velreassignment* sets the number of steps between reassigning velocities.

VELScalefactor (f; 0.0)

This keyword indicates the amount by which the initial velocities will be scaled. The default is 0.0 and it means not to scale the velocities. The keyword could be used if one generated a Boltzmann distribution of velocities at one temperature and then decided to scale the velocities to a new temperature. It is not clear why anyone would want to do that, but isn't it comforting to know that this feature is in SPASMS should you ever find a use for it?

Watermodel (m; NONE)

Special routines exist to handle water-water solvent interactions. These routines are considerably faster than the standard nonbonded routines, and provide exact answers to the other methods. Note that these are available *only* if one specifies the *residue* or *water* option for the *pairlist* methods. Also, one must specify, using the keyword *firstwaterresidue*, which residue is to be the first water residue. Finally, the water must be stored as oxygen, followed by two hydrogens. This is the standard water format from the AMBER program. There *might* be times when users really truly want to treat systems with water as ‘general’ molecules. In that case, the user would not specify *firstwaterresidue* and would use the *residue* option. Options:

| | |
|---|---|
| *none* – | There is no predefined water type. |
| *amber* – | AMBER's version of TIP3P water will be used. This is the same as the Jorgensen *et al.* original TIP3P model, except that there is a very small Lennard-Jones term added for hydrogen-hydrogen van der Waals interactions. The reason for this is to avoid fusion of two atoms during a simulation. However, this does not seem to be a problem with realistic time step sizes and with reasonable updates of the pair lists. |
| *tip3p* – | Jorgensen's TIP3P water will be used. In the TIP3P model, the interaction is calculated as one Lennard-Jones term between the oxygens and nine charge-charge terms (pairwise interactions with |

all atoms). This is the way Jorgensen *et al.* originally derived the model.

| | |
|---|---|
| *tip4p–* | Jorgensen's TIP4P water will be used. This model uses the same framework as TIP3P. However, the charge from the O atom is displaced along the bisector of the H-O-H angle. In the current implementation, all HYDROGEN bonds to or from the water are ignored. This will be corrected in the next release. Simulations with *tip4p* require no more computer time than do simulations with *tip3p*. Switches are supported. |

| | |
|---|---|
| Xbox | (f; 0.0) |
| Ybox | (f; 0.0) |
| Zbox | (f; 0.0) |

These keywords indicate the size of a periodic boundary condition box (X, Y, and Z Cartesian dimensions (angstrom)). By default, the box sizes will be read from the parameter topology file. If, for some reason, the user so desires, the defaults may be overridden here. If all three of these box lengths are non-zero, the box dimensions will be set to these values. These will *not*, however, override the box size from the INPCRD file, as those are usually read for a job continuation.

# SECTION 7: SPASMS' QUICK REFERENCE GUIDE

## 7.1 OVERVIEW

This condensed documentation is provided for users who are already familiar with the SPASMS' input (MDIN) file keywords. The keywords are arranged below by argument type and only the default keyword value, a brief keyword description, and the keyword units are listed. For more detailed information, the user should refer to *SECTION 6: DETAILED DESCRIPTION OF SPASMS' INPUT*.

## 7.2 KEYWORDS WHICH TAKE FLOATING POINT ARGUMENTS

| | | | |
|---|---|---|---|
| BETa | (90.d0) | - | angle between the (non-)orthogonal walls of a PBC box; in SPASMS, BETa should be set to 90.d0 as the option for noncubic boxes is not yet implimented |
| BUFFERRegion | (0.d0) | - | the radius (angstrom) of the buffer region |
| CAPForceconstant | (5.d-1) | - | the cap constraint force constant |
| CODrop | (0.d0) | - | the energy drop at the cutoff when using a shifted potential |
| COMpressibility | (44.6d0) | - | the inverse of the compressibility of the system (10$^{-6}$Bar$^{-1}$) |
| CONvergencecriteria | (1.d-1) | - | convergence criterion for minimization |
| CUtoff | (8.d0) | - | the non-bonded cutoff radius (angstrom) |
| DELTALambda | (0.0d0) | - | window size for free energy perturbation |
| DIelectricconstant | (2.d0) | - | the dielectric constant |
| DT | (1.d-3) | - | the time step size (psec) for the simulation |
| LAMBDAInitial | (0.d0) | - | the initial value of lambda |
| MINOrcollfrequency | (1.d-1) | - | the frequency of stochastic collisions per time step for temperature control |
| PREference | (1.d0) | - | the external pressure (Bar) on the system during a constant temperature run |
| RBondtolerance | (1.d-4) | - | the bond length tolerance (angstrom) for RATTLE |
| RDFLower | (3.d0) | - | the lower limit for computing radial distribution functions |
| RDFUpper | (12.d0) | - | the upper limit for computing radial distribution functions |
| REFtime | (0.d0) | - | the starting (reference) time of the simulation (psec) |
| RLower | (0.d0) | - | the lower limit for the switch function |
| RUPper | (0.d0) | - | the rupper limit for the switch function |
| RVelocitytolerance | (1.d-4) | - | the tolerance for velocities along constrained bonds in RATTLE |
| SCEe | (2.d0) | - | the scaling factor for the 1,4 electrostatic interactions |

| | | |
|---|---|---|
| SCNb | (2.d0) | - the scaling factor for the 1,4 VDW interactions |
| TAUPressure | (2.d-1) | - the pressure coupling constant (psec) |
| TAUTemperature | (2.d-1) | - the temperature coupling constant (psec) |
| TEmpi | (300.d0) | - the initial temperature (Kelvin) of the system |
| TFluctuation | (10.d0) | - the maximum allowed temperature variation (Kelvin) during a constant temperature run |
| TIMelimit | (99999.d0) | - the time limit for the job in CPU seconds |
| TReference | (300.d0) | - the temperature (Kelvin) of the heat bath during a constant temperature run |
| VELScalefactor | (0.d0) | - multiplier for all the velocities |
| Xbox | (0.d0) | - the X-dimension of the periodic box (angstrom) |
| Ybox | (0.d0) | - the Y-dimension of the periodic box (angstrom) |
| Zbox | (0.d0) | - the Z-dimension of the periodic box (angstrom) |

## 7.3 KEYWORDS WHICH TAKE INTEGER ARGUMENTS

| | | |
|---|---|---|
| BUFFERUpdatefrequency | (1) | - the number of steps between updating the buffer region |
| CAPAtompointer | (0) | - the index of the first atom in the cap |
| EDumpfrequency | (9999) | - the number of steps between writing to the energy file |
| FEPEQuilibrate | (0) | - equilibration steps for free energy perturbation |
| Firstwaterresidue | (0) | - the index of the first water residue |
| MAxrattleiterations | (1000) | - maximum number of RATTLE iterations per step |
| NBupdatefrequency | (1) | - the number of steps between updating the pair list |
| NDfmin | (6) | - the number of degrees of freedom to remove from the system |
| PRTfrequency | (9999) | - the number of steps between writing to the output file |
| Qdumpfrequency | (9999) | - the number of steps between writing to the coordinate file |
| RDFFrequency | (9999) | - the number of steps between collecting RDF data |
| RMcomminterval | (999999) | - the number of steps between removing the center of mass motion |
| RUNs | (0) | - the number of runs in this job |
| SEed | (71277) | - the random number seed |
| STeps | (0) | - the number of MD steps per run in this job |
| USER[1-5] | (0) | - unassigned; these are for user routines |
| VDumpfrequency | (9999) | - the number of steps between writing to the velocity file |

| | | |
|---|---|---|
| VELReassignment | (9999) | - the number of steps between reassigning velocities |

## 7.4 KEYWORDS WHICH AFFECT BOOLEAN VALUES

The default value is set unless the keyword is specified:

| | | |
|---|---|---|
| BELly | (FALSE) | - flag for a belly run |
| DDd | (TRUE) | - type of dielectric function to be used: distance dependent [‘false’]; constant: [‘true’] |
| DDRIVE or DIHEDRALDRIVER | (FALSE) | - flag for use of the dihedral driver option. Implies input to be read from file MDDRV. |
| DIHEDRALDUMP | (FALSE) | - flag indicates a coordinate file is to be dumped at the end of each dihedral driver pass. |
| DIFfrescutoff | (FALSE) | - flag for applying different non-bonded cutoffs for different residues |
| DOntremovecomm | (TRUE) | - flag to specify *not* to remove the center of mass motion; the default is to remove the center of mass motion |
| ENDofinput | ( ) | - keyword to terminate reading free form input; to specify belly and/or constraint groups terminate the free format input with the ENDofinput keyword and follow it with ‘group’ specifications |
| ENSEMbledynamics | (FALSE) | - flag to use the ensemble dynamics code. Requires input of a set of groups following the END line. |
| FLATwell | (FALSE) | - flag to use the nmr constraints on interatomic pairs and dihedral angle constraints. |
| Genpairlist | (FALSE) | - flag to generate a non-bond pair list |
| MINImization | (FALSE) | - flag for a minimization |
| MULTIPLEIN | (FALSE) | - if set, means that many input files will be read for one topology file. |
| MULTIPLEOUT | (FALSE) | - if set, many output files will be generated from one simulation. |
| OMITBondinteractions | (FALSE) | - flag to ignore all bond interactions |
| OMITHinteractions | (FALSE) | - flag to ignore bond interactions involving hydrogen |
| REDefineimagingatoms | (FALSE) | - flag to redefine imaging atoms for residues |
| RESTArt | (FALSE) | - flag for a restart |
| RESTRaints | (FALSE) | - flag for applying coordinate constraints |
| SETTLE | (FALSE) | - flag for applying settle to water molecules |
| USEPDBIN | (FALSE) | - flag to extract coordinates from an input pdb file |

| | | | |
|---|---|---|---|
| USEPDBOUT | (FALSE) | - | flag to use pdb format on output of coordinate files |
| USEPDBDUMP | (FALSE) | - | flag to use pdb format on dumping of coordinate files |
| USEPDBDRIVEDUMP | (FALSE) | - | flag to use pdb format on dumping dihedral driver coordinate files |

## 7.5 KEYWORDS WHICH TAKE COMPLEX ARGUMENTS OR SET MANY OPTIONS

| | | | |
|---|---|---|---|
| CAPOption | (INACTIVE) | - | specification of the cap option; options: {ACTIVE – the cap will be in effect if specified in the PRMTOP file, MODIFY – the cap will be in effect and the cap atom index will be modified, INACTIVE – the cap will not be in effect} |
| COOrdinateinput | (CRD) | - | specification of what is read from the input coordinate [INPCRD] file; options: {CRD – initial Cartesian coordinates are read, VEL – initial velocities are read, BOX – initial box dimensions are read, REFC – reference Cartesian coordinates are read} |
| ROOTDRIVE | (ROOTDRIVE) | - | the rootname for files generated at the end of each dihedral driver pass. Ignored if DRIVEDUMP is false. |
| DRIVEEXTENSION | (DRV) | - | the extension for files generated at the end of each dihedral driver pass. |
| Initvelocities | (RANDOM) | - | the source of the initial velocities; options: {RANDOM – the initial velocities are assigned randomly from a Boltzman distribution, INPUT – the initial velocities are read from the inpcrd file} |
| ROOTNAMEIN<br>ROOTNAMEOUT<br>INEXTENSION<br>OUTEXTENSION<br>ROOTNAMEDUMP | | - | rootnames and extensions for the multiple input files, the multiple output files and the multiple coordinate dump files. |
| NPT | ( ) | - | specify a run using the NPT ensemble; temperature control is achieved using stochastic collisions |
| NVE | ( ) | - | specify a run using the micro-canonical ensemble; there is no temperature control |
| NVT | ( ) | - | specify a run using the canonical ensemble; temperature control is achieved using stochastic collisions |

| | | | |
|---|---|---|---|
| PAirlist | (ATOM) | - | specification of the type of pair list to generate; options: {RESIDUE – generate a residue-based pair list, ATOM – generate an atom-based pair list, FIRSTATOM– generate a pair list based on the first atoms in each residue, SPECIALATOM – use the special atoms for determining the pairlist, SHIFT – use a shifted potential with an atom-based list} |
| PBc | (NONE) | - | specification of what kind of periodic boundary condition to use; options: {NONE – no periodic boundary condition is applied, VOLUME – perform a constant volume simulation, PRESSURE – perform a constant pressure simulation} |
| PControl | (NONE) | - | specification of the pressure control (coupling) scheme; options: {NONE – no pressure coupling is employed, ISOTROPIC – the box dimensions change isotropically to maintain pressure, ANISOTROPIC – the box dimensions change anisotropically to maintain pressure} |
| RAttle | (NONE) | - | specification of the RATTLE choice; options: {NONE – no bonds are RATTLE'd, HYDROGEN – all bonds involving hydrogen are RATTLE'd, ALL – all bonds are RATTLE'd} |
| RDFOption | ( ) | - | specification of collecting radial distribution function data; options: {default is no collection; the RDF option specification requires three parameters – the frequency, lower, and upper bounds; if these are specified using the RDFOption keyword, the must use the syntax: RDFOption = (<rdflower>, rdfupper>, <rdffreq>); alternately, RDFOption can be specified with no arguments and rdflower, rdfupper, and rdffreq can be specified using the RDFLower, RDFUpper, and RDFfreq keywords} |
| SWitchoption | ( ) | - | specification of the switch function; options: {the default is no switch function; the switch function specification requires two parameters – 'rlower' and 'rupper'; if these are specified using the SWitchoption keyword, then the syntax: SWitchoption = (<rlower>,<rupper>) must be used; alternately, SWitchoption can be |

| | | | |
|---|---|---|---|
| | | | specified with no arguments and rlower and rupper are specified using the RLower and RUPper keywords} |
| TControl | (NONE) | - | specification of the temperature control (coupling) scheme; options: {NONE – no temperature coupling is employed, VELOCITY – temperature control is accomplished by rescaling velocities, RANDOM – temperature control is accomplished by stochastic collisions} |
| TITle | ( ) | - | title of the run; options: {the allowed option is any string; if the string contains more than one word, it must be in quotes} |
| Watermodel | (NONE) | - | specification of which water model to use if using fast water routines; options: {NONE – no predefined water type, AMBER – use AMBER'S version of TIP3P water, TIP3P – use Jorgensen's TIP3P water, TIP4P – use Jorgensen's TIP4P water model. Note: the hydrogen bonds to and from the water are eliminated in this model. SHAKE/RATTLE or SETTLE must be set to use TIP3P or TIP4P water models.} |