NAMD User Guide

Version 1.5

R. Brunner, A. Dalke, A. Gursoy, W. Humphrey, M. Nelson

September 4, 1998

Theoretical Biophysics Group
University of Illinois and Beckman Institute
405 N. Mathews
Urbana, IL 61801

Description

The NAMD *User Guide* describes how to run and use the various features of the molecular dynamics program NAMD. This guide includes the capabilities of the program, how to use these capabilities, the necessary input files and formats, and how to run the program both on uniprocessor machines and in parallel.

NAMD Version 1.5

Authors: Robert Brunner, Andrew Dalke, Attila Gursoy,
Bill Humphrey, and Mark Nelson

Theoretical Biophysics Group, Beckman Institute, University of Illinois.

NOTICE

The program NAMD is *not* in the public domain. However, it is freely available without fee for education, research, and commercial purposes. By obtaining copies of this and other files that comprise the NAMD program, you, the Licensee, agree to abide by the following conditions and understandings with respect to the copyrighted software:

1. The software is copyrighted in the name of the Board of Trustees of the University of Illinois (UI), and ownership of the software remains with the UI.

2. Permission to use and modify this software and its documentation is hereby granted to Licensee. In addition, permission to copy this work and any derived works is granted to Licensee, provided that

   (a) the copyright notice and this permission notice appear on all such copies,

   (b) that proper credit be given by citing

      M. Nelson, W. Humphrey, A. Gursoy, A. Dalke, L. Kale, R. Skeel and K. Schulten. NAMD - A parallel, object-oriented molecular dynamics program. *J. Supercomputing App.*, 10:251-268, 1996.

   for NAMD and

      W. T. Rankin and J. A. Board, Jr., Tech Rept. 95-002, EE Dept., Duke Univ.

   for DPMTA

   (c) that NAMD and any derived works are not redistributed for a fee,

   (d) that programs derived from NAMD be given a different name.

3. Licensee may not use the name, logo, or any other symbol of the UI nor the names of any of its employees nor any adaptation thereof in advertizing or publicity pertaining to the software without specific prior written approval of the UI.

4. THE UI MAKES NO REPRESENTATIONS ABOUT THE SUITABILITY OF THE SOFTWARE FOR ANY PURPOSE. IT IS PROVIDED "AS IS" WITHOUT EXPRESS OR IMPLIED WARRANTY.

5. The UI shall not be liable for any damages suffered by Licensee from the use of this software.

Contents

List of Figures

1 Introduction

NAMD is a parallel molecular dynamics program for UNIX platforms designed for high-performance simulations in structural biology. This document is divided into several sections:

Section 1 gives an overview of NAMD.

Section 2 discusses installing and running NAMD.

Section 3 explains the input and output file formats.

Section 4 describes the NAMD configuration file options.

Section 5 presents the details of what NAMD computes.

For those that wish to modify the NAMD source code, there is a companion document, the NAMD *Programming Guide*, that explains the design and implementation of NAMD. However, the NAMD *User Guide* should be sufficient for those that wish to use NAMD without modifying the source.

We have attempted to make both the NAMD *User Guide* and the NAMD *Programming Guide* complete and easy to understand and to make NAMD itself easy to install and run. Please take a moment to help us improve the documentation and the code by filling out the user survey at the end of this section.

1.1 New features in version 1.5

There are two new features in NAMD version 1.5: rigid bonds and moving harmonic restraints.

NAMD allows rigid water, with rigid bond angle and bond lengths, and also rigid bonds between hydrogen and heavy atoms. The SHAKE algorithm is used to correct the position so that the center of mass of the bonded atoms does not move, and the RATTLE algorithm is used to correct the velocity so that the kinetic energy is computed correctly. See section 5.2.5 for more details on the algorithms and page 26 for the controlling parameters.

NAMD provides moving harmonic restraints that allow steered molecular dynamics (SMD) simulations. This feature allows a harmonic restraining force to be applied to a single atom in the simulation and the restraining reference position to be moved. See section 5.1.9 for an explaination of the changes to the force field and page 29 for the controlling parameters.

There have been a number of bug fixes and performance improvements since the previous version. The make file structure has been changed so that NAMD is hopefully easier to build and install. In addition, NAMD now works with DPMTA version 2.7, and the DPMTA source code is distributed with NAMD and built automatically.

1.2 New features in version 1.4

In addition to bug fixes, there are two new features in NAMD version 1.4: a safety margin violation check and a new switching function.

NAMD now checks for safety margin violations at each timestep. Each atom's position is compared to its position at the beginning of the current cycle, and if the atom has moved a distance which exceeds half of the safety margin, then a report is made. This new feature alerts the user of a possible degradation in accuracy. It can be enabled or disabled with the configuration parameter `plMarginCheck` (page 24). The default is to have checking disabled.

The new switching function is obtained by setting the configuration parameter `longSplitting` to `c1`. It implements the switching method described in [7].

1.3 NAMD and molecular dynamics simulations

Molecular dynamics (MD) simulations compute atomic trajectories by solving equations of motion numerically using empirical force fields, such as the CHARMm force field, that approximate the actual atomic force in biopolymer systems. Detailed information about MD simulations can be found in several books such as [1, 8]. In order to conduct MD simulations, various computer programs have been developed including X-PLOR [6] and CHARMm [5]. These programs were originally developed for serial machines. Simulation of large molecules, however, require enormous computing power. One way to achieve such simulations is to utilize parallel computers. In recent years, distributed memory parallel computers have been offering cost-effective computational power. NAMD was designed to run efficiently on such parallel machines for simulating large molecules.

NAMD has several important features:

- **Force Field Compatibility**
  The force field used by NAMD is the same as that used by the programs CHARMm [5] and X-PLOR [6]. This force field includes local interaction terms consisting of bonded interactions between 2, 3, and 4 atoms and pairwise interactions including electrostatic and van der Waals forces. This commonality allows simulations to migrate between these three programs.

- **Fast Multipole Algorithm**
  NAMD incorporates the Distributed Parallel Multipole Tree Algorithm (DPMTA) [4], which takes the full electrostatic interactions into account. This algorithm reduces the computational complexity of electrostatic force evaluation from $O(N^2)$ to $O(N)$.

- **Multiple Time Stepping:**
  The velocity Verlet integration method [1] is used to advance the positions and velocities of the atoms in time. To further reduce the cost of the evaluation of long range electrostatic forces, a multiple time step scheme is combined with the DPMTA method. The local interactions (bonded interactions and electrostatic interactions within a specified distance) are calculated at each time step. The longer range interactions (electrostatic interactions beyond the specified distance) are only computed every k steps, with each set of k steps called a *cycle*. This amortizes the cost of computing the electrostatic forces over the k steps in the cycle. For appropriate values of k, the error due to holding the long range forces constant for a few steps is modest compared to the errors incurred from using a finite time step for the other forces.

- **Input and Output Compatibility**
  The input and output file formats used by NAMD are identical to those used by X-PLOR. Input formats include coordinate files in PDB format [2], structure files in PSF format, and energy parameter files in the same format used by CHARMm and X-PLOR. Output formats include PDB coordinate files and binary DCD trajectory files. These similarities assure that the molecular dynamics trajectories from NAMD can be read by CHARMm or X-PLOR and that the user can exploit the many analysis algorithms of the latter packages.

- **Dynamics Simulation Options**
  MD simulations may be carried out using several options, including
  - NVE ensemble dynamics,
  - NVT ensemble dynamics,
  - Velocity rescaling,
  - Langevin dynamics,
  - Energy minimization,
  - Harmonic restraints,
  - Moving harmonic restraints,
  - Spherical boundary conditions,
  - Rigid bonds.
- **Easy to Modify and Extend**
  Another primary design objective for NAMD is extensibility and maintainability. In order to achieve this, it is designed in an object-oriented style with C++. Since molecular dynamics is a new field, new algorithms and techniques are continually being developed. NAMD's modular design allows one to integrate and test new algorithms easily. The structure and design of the program is described in the NAMD *Programming Guide* with sufficient detail to allow additions of such new algorithms.
- **Interactive MD simulations**
  NAMD is one component of MDScope, a computational environment for structural biology. The two other components of MDScope are VMD, a molecular visualization program, and MDComm, a communications package. When combined, a system undergoing simulation in NAMD may be viewed and altered with VMD; for instance, forces can be applied to a set of atoms to alter or rearrange part of the molecular structure. For more information on MDScope, see `http://www.ks.uiuc.edu/Research/mdscope/`.
- **Load Balancing**
  An important factor in parallel applications is the equal distribution of computational load among the processors. In parallel molecular simulation, a spatial decomposition that evenly distributes the computational load causes the region of space mapped to each processor to become very irregular, hard to compute and difficult to generalize to the evaluation of many different types of forces. NAMD addresses this problem by using a simple uniform spatial decomposition where the entire model is split into uniform cubes of space called *patches*. An initial load balancer assigns patches to processors such that the computational load is balanced as much as possible. During the simulation, an incremental load balancer monitors the load and performs necessary adjustments.

1.4 User feedback

If you have problems installing or running NAMD after reading this document, please send a complete description of the problem by email to `namd@ks.uiuc.edu` . If you discover and fix a problem not described in this manual or the NAMD *Programming Guide*, we would appreciate if you would tell us about this as well, so we can alert other users and incorporate the fix into the public distribution.

We are interested in making NAMD more useful to the molecular modeling community. Please take a few minutes to complete the short survey on the next page and mail or fax it to

NAMD Support
Theoretical Biophysics Group
Beckman Institute
University of Illinois
405 N. Mathews St.
Urbana, IL 61801
FAX (217) 244-6078

or email the relevant information to `namd@ks.uiuc.edu` with subject "survey." This will help us make NAMD relevant and responsive to the needs of our user community.

1. Name:
2. Company/University:
3. Address:
4. Email address:
5. Research area:
6. Molecular systems to be studied with NAMD:
7. NAMD features most useful for your research:
8. I run NAMD on the following platforms and operating systems:
9. I would like to run NAMD on:
10. How can we improve the NAMD documentation?
11. Other comments:

1.5 Acknowledgments

This work is supported by grants from the National Science Foundation (BIR-9318159) and the National Institute of Health (PHS 5 P41 RR05969-04).

The authors would particularly like to thank the members of the Theoretical Biophysics Group, past and present, who have helped tremendously in making suggestions, pushing for new features, and testing bug-ridden code.

2 Installing and Running

2.1 Obtaining NAMD

NAMD source and documentation is available in the compressed archive file `namd-1.5.tar.gz` obtained via anonymous FTP from `ftp.ks.uiuc.edu` in the directory `pub/mdscope/namd` .

2.2 Supported platforms

NAMD is designed to run as a distributed application under UNIX with communication performed by PVM. The source code is written entirely in C and C++. We have successfully ported NAMD to all of our UNIX machines, which include HP, SGI, Sun, and Linux, both single processor and shared memory multiprocessor.

There is one note of concern. Performance has been poor on our dual processor Linux cluster. The problem seems to be a result of the number of system calls that are made from the PVM library from the `namd` process (over 50000 calls to `gettimeofday` and over 30000 calls to `select` in 8 timesteps, mostly from the function `mxfer()` in `$PVM_ROOT/src/lpvm.c`). We are not sure whether this is a problem with PVM under Linux or with the way in which NAMD is using PVM. We have tested using PVM versions 3.3.11 and 3.4 on RedHat Linux distributions Manhattan and Wiglaf. The problems are only evident when using more than one node. NAMD performs fine on a single node. We will continue to examine this problem for possible solutions, but any help is welcome.

The build procedure has been overhauled to make it easier to target new architectures and support multiple platforms from a single file system. Hopefully, it will also make installation easier. Now, the only two files that need to be modified are:

- `Make.config` — provides platform independent compile options
- `Make.arch` — provides compile options for specific architectures

These files are well-documented as to what should and should not be modified. For reference, the make files from the previous distribution have been placed in the 'OLD.Makefiles/' subdirectory. Although the DPMTA directories have been modified from their original distribution, the original DPMTA make files have also been retained as 'OLD.filename'.

2.3 Required packages

NAMD currently runs on PVM (Parallel Virtual Machine), a message-passing library available from Oak Ridge National Laboratory. PVM runs on many distributed and parallel platforms, and the source is freely available through `http://www.epm.ornl.gov/pvm/pvm_home.html`. You must have PVM installed on your system prior to building NAMD. This version of NAMD has been developed and tested using PVM 3.3.11 and 3.4.

At present, there is no good way to generate the required PSF files without some additional package. We recommend X-PLOR (`http://xplor.csb.yale.edu/xplor-info/xplor-info.html`) to generate these structure files.

2.4 Optional packages

The DPMTA (Distributed Parallel Multipole Treecode Algorithm) package from the Scientific Computing Group at Duke University (`http://www.ee.duke.edu/Research/SciComp/SciComp.html`). provides efficient full electrostatic calculations using a variation of the FMA algorithm. The source code for version 2.7 has been included in this distribution of NAMD to ease the installation of both packages and to ensure compatibility as development on DPMTA continues. The package is optional in the sense that you can perform cutoff simulations without DPMTA by specifying the configuration file parameter `FMA` to be `off`. If you do *not* want DPMTA support, simply comment out the following variable assignments in `Make.config`:

- `NAMD_CFLAGS_DPMTA`
- `NAMD_LIBS_DPMTA`

and change the `Make.config` `BUILD` variable to:

- `BUILD = build_namd install`

Otherwise, DPMTA will be built automatically with NAMD.

MDComm (`http://www.ks.uiuc.edu/Research/mdcomm/`) is a library routine and set of daemon processes that allow NAMD to communicate with the molecular visualization program VMD (`http://www.ks.uiuc.edu/Research/vmd/`) running on a remote machine via TCP/IP. By default, this package is not compiled with NAMD. To use with NAMD, first install MDComm and then uncomment the appropriate lines in `Make.config` and indicate the absolute path. However, please note that MDComm is presently unsupported.

2.5 Compiling and installing NAMD

Follow these steps in compiling NAMD:

1. Make sure that PVM is installed correctly and that the libraries are present for the architecture you wish to compile NAMD. You should have the `PVM_ROOT` variable defined in your environment.

2. Edit the files `Make.config` and `Make.arch` as needed. The `Make.config` file contains architecture independent information, such as path information and general compile flags and libraries. Very few changes should be necessary. The `Make.arch` file contains specific architecture targets and lets you indicate compilers and compile flags. You should use the name of the target in `Make.arch` as the name of the subdirectory under which all object files, library files, and executables for this build will be stored. If your particular architecture is not listed, you will need to add another target and define the listed variables. See the `Make.config` and `Make.arch` files under "What should be changed?" for more details.

3. To make NAMD for some *target* architecture listed in `Make.arch`, type:

   - `make` *target*

   If older object files exist from a previous build, you can first clean all of the binaries for a given *target* architecture by typing:

- `make clean "ARCH=`*target*`"`

4. The build should automatically install links to the `namd` and `dpmta_slave` binaries in the indicated `PVM_BINDIR` directory. If you wish to install links to a different *target* build into a given *pvm_target* directory, you can install NAMD manually by typing:

   - `make install "ARCH=`*target*`" "PVM_ARCH=`*pvm_target*`"`

2.6 Running NAMD simulations

NAMD requires certain input files that specify the system coordinates and molecular structure and a configuration file that lists simulation options. See Sections 3 and 4 for more details concerning these files.

In a distributed simulation with full electrostatics using DPMTA, there will be three processes running per node:

- `namd` — the NAMD process
- `dpmta_slave` — the DPMTA slave process
- `pvmd` — the PVM daemon process

Suppose we wish to start a distributed NAMD simulation on, say, the four hosts `foo1`, `foo2`, `foo3`, and `foo4`. From host `foo1` we could invoke the PVM daemon and add the other hosts, execute NAMD using configuration file `my-config-file`, and halt PVM after NAMD exits. The sequence of commands looks like:

```
foo1> pvm
pvm> add foo2 foo3 foo4
3 successful

...  more output from PVM appears here  ...

pvm> quit

pvmd still running.
foo1> namd my-config-file

...  a whole lot of output from NAMD appears here  ...

foo1> pvm
pvmd already running.
pvm> halt
foo1>
```

Now, suppose we wish to start a multi-processor NAMD simulation on, say, the host `foo`. From host `foo` we invoke the PVM daemon, execute NAMD using configuration file `my-config-file` along with the number of nodes to use, and halt PVM after NAMD exits. (Note that NAMD needs to have been compiled with the `-DSPECIFY_NUM_NODES` option on the `NAMD_CFLAGS_ARCH` line in `Make.arch`.) The sequence of commands looks like:

```
foo> pvm
pvm> quit

pvmd still running.
foo> namd my-config-file 4

...  a whole lot of output from NAMD appears here  ...

foo> pvm
pvmd already running.
pvm> halt
foo>
```

It might also be necessary to set the environmental variable `PVMBUFSIZE` that indicates the amount of space to devote to message buffering. We suggest anywhere from 4 to 32 MB depending on the size of the molecular system. See the PVM documentation for more details.

2.7 Testing NAMD

There are sample configuration files in the `sample/` directory. After building NAMD, change to the `sample/` directory and invoke NAMD on each of the configuration files. (Don't forget to start PVM first!) Assuming that the NAMD executable is in your path, perform the first test by typing:

```
> namd alanin1.config
```

The last output line of energies should be approximately:

```
Info> ETITLE:      TS    BOND        ANGLE       DIHED       IMPRP       ELECT
    VDW       KINETIC       TOTAL       TEMP        |P|         |L|
Info> ENERGY:    1000 1.7517      8.8425      1.0274      2.4743      -207.3765
 -29.9398    15.0133     -208.2070  77.4878    4.3095e-14 2.1252e-03
```

Then perform the second test by typing:

```
> namd alanin2.config
```

The last output line of energies should be approximately:

```
Info> ETITLE:      TS    BOND        ANGLE       DIHED       IMPRP       ELECT
    VDW       KINETIC       TOTAL       TEMP        |P|         |L|
Info> ENERGY:     100 0.2072      4.7894      0.8302      3.5955      -272.3828
 -30.9772    13.9564     -279.9814  72.0327    7.6971e-02 6.4902e-01
```

Note that using FMA on only 66 atoms is expected to be substantially slower than computing full electrostatics using the direct $\mathcal{O}(N^2)$ approach.

3 Molecular Input and Output File Formats

NAMD was developed to be compatible with existing molecular dynamics packages, especially the packages X-PLOR [6] and CHARMm [5]. To achieve this compatibility, the set of input files which NAMD uses to describe a molecular system are identical to the input files used by X-PLOR. These input files are very similar to those used by CHARMm. Thus it is trivial to move an existing simulation from X-PLOR to NAMD and requires minimal effort to convert a simulation from CHARMm to NAMD. In addition, the output file formats used by NAMD were chosen to be compatible with X-PLOR and CHARMm. In this way the output from NAMD can be analyzed using X-PLOR, CHARMm, or a variety of the other tools that have been developed for the existing output file formats.

Besides these input and output files, NAMD also uses a file referred to as the *configuration file*. This file specifies what dynamics options and values that NAMD should use, such as the number of timesteps to perform, initial temperature, etc. The options and values in this file control how the system will be simulated. The format and options used in this configuration file are described in Section 4.

3.1 PDB files

The PDB (Protein Data Bank) format is used to store coordinate or velocity data being input or output from NAMD. This is the standard format for coordinate data for most other molecular dynamics programs as well, including X-PLOR and CHARMm. A full description of this file format can be obtained via anonymous FTP from `ftp.pdb.bnl.gov` in `/pub/format.desc.ps.Z` or `/pub/format.desc.txt`.

3.2 X-PLOR format PSF files

NAMD uses the same protein structure files that X-PLOR does. At this time, the easiest way to generate these files is using X-PLOR, although it is possible to build them by hand.

3.3 CHARMm19 and CHARMm22 parameter files

NAMD supports CHARMm19 and CHARMm22 parameter files in the same format that is used by X-PLOR. For a full description of the format of commands used in these files, see the X-PLOR User's Manual [6].

3.4 DCD trajectory files

NAMD produces DCD trajectory files in the same format as X-PLOR and CHARMm. The DCD files are single precision binary FORTRAN files, so are portable between computer architectures. The exact format of these files is very ugly but supported by a wide range of analysis and display programs.

4 NAMD Configuration File

A NAMD configuration file contains a set of options and values. The options and values specified determine the exact behavior of NAMD, what features are active or inactive, how long the simulation should continue, etc. Section 4.1 describes how options are specified within a NAMD configuration file. Section 4.2 describes the various parameters that are available and how to use them. Section 4.3 describes the relation between specific NAMD and X-PLOR dynamics options. Several sample NAMD configuration files are shown in section 4.4

4.1 Configuration parameter syntax

Each line in the configuration files consists of a *keyword* identifying the option being specified, and a *value* which is a parameter to be used for this option. The keyword and value can be separated by only white space:

```
keyword            value
```

or the keyword and value can be separated by an equal sign and white space:

```
keyword      =     value
```

Blank lines in the configuration file are ignored. Comments are prefaced by a `#` and may appear on the end of a line with actual values:

```
keyword            value          #  This is a comment
```

or may be at the beginning of a line:

```
#  This entire line is a comment . . .
```

The specification of the keywords is case insensitive so that any combination of upper and lower case letters will have the same meaning. Hence, `DCDfile` and `dcdfile` are equivalent. The capitalization in the values, however, may be important. Some values indicate file names, in which capitalization is critical. Other values such as `on` or `off` are case insensitive.

4.2 NAMD configuration parameters

This section describes the various configuration parameters that are available with NAMD. It is broken into subsections that group similar parameters together. The following parameters are *required* for every NAMD simulation:

- `numsteps` (page 22),
- `coordinates` (page 19),
- `structure` (page 19),
- `parameters` (page 19),

- `exclude` (page 24),
- `outputname` (page 20),
- either `cutoff` (page 23) or both of `eleccutoff` (page 23) and `vdwcutoff` (page 23),
- one of the following three:
  - `temperature` (page 25),
  - `velocities` (page 20),
  - `binvelocities` (page 20).

These required parameters specify the most basic properties of the simulation. In addition, it is highly recommended that

- `pairlistdist` be specified with a value at least one greater than the maximum of `cutoff`, `vdwcutoff`, and `eleccutoff`,
- `stepspercycle` be chosen so that the product of `stepspercycle` and `timestep` does not exceed 4.0 unless `rigidBonds all` is specified, in which case the upper limit is perhaps doubled.

Input files

- `coordinates` ⟨ coordinate PDB file ⟩
  Acceptable Values: UNIX filename
  Description: The PDB file containing initial position coordinate data. Note that path names can be either absolute or relative. Only one value may be specified.
- `structure` ⟨ PSF file ⟩
  Acceptable Values: UNIX filename
  Description: The X-PLOR style PSF file describing the molecular system to be simulated. Only one value may be specified.
- `parameters` ⟨ parameter file ⟩
  Acceptable Values: UNIX filename
  Description: A CHARMM19 or CHARMM22 parameter file that defines all or part of the parameters necessary for the molecular system to be simulated. At least one parameter file must be specified for each simulation. Multiple definitions are allowed for systems that require more than one parameter file. For example, if three files were needed, lines such as:

  ```
  parameters param1
  parameters param2
  parameters param3
  ```

  could be added to the configuration file. The files will be read in the order that they appear in the configuration file. If duplicate parameters are read, a warning message is printed and the last parameter value read is used. Thus, the order that files are read can be important in cases where duplicate values appear in separate files.

- `velocities` ⟨ velocity PDB file ⟩
  Acceptable Values: UNIX filename
  Description: The PDB file containing the initial velocities for all atoms in the simulation. This is typically a restart file or final velocity file written by NAMD during a previous simulation. Either the `temperature` or the `velocities/binvelocities` option must be defined to determine an initial set of velocities. Both options cannot be used together.

- `binvelocities` ⟨ binary velocity file ⟩
  Acceptable Values: UNIX filename
  Description: The binary file containing initial velocities for all atoms in the simulation. A binary velocity file is created as output from NAMD by activating the `binaryrestart` option. The `binvelocities` option should be used as an alternative to `velocities`. Either the `temperature` or the `velocities/binvelocities` option must be defined to determine an initial set of velocities. Both options cannot be used together.

- `bincoordinates` ⟨ binary coordinate restart file ⟩
  Acceptable Values: UNIX filename
  Description: The binary restart file containing initial position coordinate data. A binary coordinate restart file is created as output from NAMD by activating the `binaryrestart` option. Note that, in the current implementation at least, the `bincoordinates` option must be used in addition to the `coordinates` option, but the positions specified by `coordinates` will then be ignored.

- `cwd` ⟨ default directory ⟩
  Acceptable Values: UNIX directory name
  Description: The default directory for input and output files. If a value is given, all filenames that do not begin with a / are assumed to be in this directory. For example, if `cwd` is set to `/scr`, then a filename of `outfile` would be modified to `/scr/outfile` while a filename of `/tmp/outfile` would remain unchanged. If no value for `cwd` is specified, than all filenames are left unchanged.

Output files

- `outputname` ⟨ output PDB file ⟩
  Acceptable Values: UNIX filename prefix
  Description: At the end of every simulation, NAMD writes two PDB files, one containing the final coordinates and another containing the final velocities of all atoms in the simulation. This option specifies the file prefix for these two files. The position coordinates will be saved to a file named as this prefix with `.coor` appended. The velocities will be saved to a file named as this prefix with `.vel` appended. For example, if the prefix specified using this option was `/tmp/output`, then the two files would be `/tmp/output.coor` and `/tmp/output.vel`.

- `restartname` ⟨ restart files ⟩
  Acceptable Values: UNIX filename prefix
  Description: The prefix to use for restart filenames. NAMD produces PDB restart files that store the current positions and velocities of all atoms at some step of the simulation. This option specifies the prefix to use for restart files in the same way that `outputname` specifies a filename prefix for the final positions and velocities. If `restartname` is defined then the parameter `restartfreq` must also be defined.

- `restartfreq` ⟨ frequency of restart file generation ⟩
  Acceptable Values: positive integer
  Description: The number of timesteps between the generation of restart files. If `restartfreq` is defined, then `restartname` must also be defined.

- `binaryrestart` ⟨ use binary restart files? ⟩
  Acceptable Values: `yes` or `no`
  Default Value: `yes`
  Description: Activates the use of binary restart files. If this option is set to `yes`, then the restart files and final output files will be written in binary rather than PDB format. Binary files preserve more accuracy between NAMD restarts than ASCII PDB files, but the binary files are not guaranteed to be transportable between computer architectures.

- `DCDfile` ⟨ coordinate trajectory output file ⟩
  Acceptable Values: UNIX filename
  Description: The binary DCD position coordinate trajectory filename. This file stores the trajectory of all atom position coordinates using the same format (binary DCD) as X-PLOR. If `DCDfile` is defined, then `DCDfreq` must also be defined.

- `DCDfreq` ⟨ timesteps between writing coordinates to trajectory file ⟩
  Acceptable Values: positive integer
  Description: The number of timesteps between the writing of position coordinates to the trajectory file.

- `velDCDfile` ⟨ velocity trajectory output file ⟩
  Acceptable Values: UNIX filename
  Description: The binary DCD velocity trajectory filename. This file stores the trajectory of all atom velocities using the same format (binary DCD) as X-PLOR. If `velDCDfile` is defined, then `velDCDfreq` must also be defined.

- `velDCDfreq` ⟨ timesteps between writing velocities to trajectory file ⟩
  Acceptable Values: positive integer
  Description: The number of timesteps between the writing of velocities to the trajectory file.

- `electForceDCDfile` ⟨ electrostatic force trajectory file ⟩
  Acceptable Values: UNIX filename prefix
  Description: The binary DCD short and long range electrostatic force trajectory filename prefix. The short and long range electrostatic force trajectories will be respectively stored in files with `.s` and `.l` appended to this prefix. If `electForceDCDfile` is defined, then `electForceDCDfreq` must also be defined.

- `electForceDCDfreq` ⟨ timesteps between writing electrostatic forces to trajectory file ⟩
  Acceptable Values: positive integer
  Description: The number of timesteps between writing to the long and short range electrostatic force trajectory files.

- `allForceDCDfile` ⟨ total force trajectory file ⟩
  Acceptable Values: UNIX filename
  Description: The binary DCD total force trajectory filename. This file stores the trajectory

of the total forces on each atom. If `allForceDCDfile` is defined, then `allForceDCDfreq` must also be defined.

- `allForceDCDfreq` ⟨ timesteps between writing total forces to trajectory file ⟩
  Acceptable Values: positive integer
  Description: The number of timesteps between the writing of total forces to the trajectory file.

- `outputEnergies` ⟨ timesteps between energy output ⟩
  Acceptable Values: positive integer
  Default Value: 1
  Description: The number of timesteps between each energy output of NAMD. This value specifies how often NAMD should output the current energy values to **stdout** (which can be redirected to a file). By default, this is done every step. For long simulations, the amount of output generated by NAMD can be greatly reduced by outputting the energies only occasionally.

Timestep parameters

- `numsteps` ⟨ number of timesteps ⟩
  Acceptable Values: positive integer
  Description: The number of simulation timesteps to be performed. An integer greater than 0 is acceptable. The total amount of simulation time is `numsteps` × `timestep`.

- `timestep` ⟨ timestep size (fs) ⟩
  Acceptable Values: non-negative decimal
  Default Value: 1.0
  Description: The timestep size to use when integrating each step of the simulation. The value is specified in femtoseconds.

- `firsttimestep` ⟨ starting timestep value ⟩
  Acceptable Values: non-negative integer
  Default Value: 0
  Description: The number of the first timestep. This value is typically used only when a simulation is a continuation of a previous simulation. In this case, rather than having the timestep restart at 0, a specific timestep number can be specified.

- `stepspercycle` ⟨ timesteps per cycle ⟩
  Acceptable Values: positive integer
  Default Value: 20
  Description: Number of timesteps in each cycle. Each cycle represents the number of timesteps between pairlist generation and atom reassignment. If full electrostatics are active, it is also the number of timesteps between full electrostatic evaluation. It is recommended that `stepspercycle` be chosen so that the product of `stepspercycle` and `timestep` does not exceed 4.0 unless `rigidBonds all` is specified, in which case the upper limit is perhaps doubled. For more details on the use of full electrostatics, see section 4.5.4. For more details on non-bonded force evaluation and pairlist generation, see section 4.5.

Simulation space partitioning

- `eleccutoff` ⟨ local interaction distance for electrostatic calculations (Å) ⟩
  Acceptable Values: positive decimal
  Description: If DPMTA is active, this distance defines the local interaction length for the DPMTA algorithm. Otherwise, this value specifies the distance at which electrostatic interactions are truncated. If `eleccutoff` is defined, it supersedes `cutoff`. If `eleccutoff` is not defined, then `cutoff` *must* be defined. See section 4.5 for a further discussion of this configuration value.

- `vdwcutoff` ⟨ local interaction distance for van der Waals calculations (Å) ⟩
  Acceptable Values: positive decimal
  Description: This value specifies the distance at which van der Waals interactions are truncated. If `vdwcutoff` is defined, it supersedes `cutoff`. If `vdwcutoff` is not defined, then `cutoff` *must* be defined. See section 4.5 for a further discussion of this configuration value.

- `cutoff` ⟨ local interaction distance common to both electrostatic and van der Waals calculations (Å) ⟩
  Acceptable Values: positive decimal
  Description: This value can substitute for either `eleccutoff` or `vdwcutoff` if either of those is undefined. See section 4.5 for more information.

- `switching` ⟨ use switching function? ⟩
  Acceptable Values: `on` or `off`
  Default Value: `off`
  Description: If `switching` is specified to be `off`, then a truncated cutoff is performed. If `switching` is turned `on`, then smoothing functions are applied to both the electrostatics and van der Waals forces. For a complete description of the non-bonded force parameters see section 4.5. If `switching` is set to `on`, then `switchdist` must also be defined.

- `elecswitchdist` ⟨ distance at which to activate switching function for electrostatic calculations (Å) ⟩
  Acceptable Values: positive decimal ≤ `eleccutoff`
  Description: Distance at which the switching function used to smooth the truncation of electrostatic forces should begin to take effect. This parameter only has meaning if `switching` is set to `on`. The value of `elecswitchdist` must be less than or equal to the value of `eleccutoff`, since the switching function is only applied on the range from `elecswitchdist` to `eleccutoff`. If `elecswitchdist` is defined, it supersedes `switchdist`. If `elecswitchdist` is not defined and `switching` is `on`, then `switchdist` *must* be defined. For a complete description of the non-bonded force parameters, see section 4.5.

- `vdwswitchdist` ⟨ distance at which to activate switching function for van der Waals calculations (Å) ⟩
  Acceptable Values: positive decimal ≤ `vdwcutoff`
  Description: Distance at which the switching function used to smooth the truncation of van der Waals forces should begin to take effect. This parameter only has meaning if `switching` is set to `on`. The value of `vdwswitchdist` must be less than or equal to the value of `vdwcutoff`, since the switching function is only applied on the range from `vdwswitchdist` to `vdwcutoff`. If `vdwswitchdist` is defined, it supersedes `switchdist`. If `vdwswitchdist` is not defined

and `switching` is `on`, then `switchdist` *must* be defined. For a complete description of the non-bonded force parameters, see section 4.5.

- `switchdist` ⟨ distance at which to activate switching function for electrostatic and van der Waals calculations (Å) ⟩
  Acceptable Values: positive decimal ≤ `cutoff`
  Description: Distance at which the switching function should begin to take effect. This parameter only has meaning if `switching` is set to `on`. The value of `switchdist` must be less than or equal to the value of `cutoff`, since the switching function is only applied on the range from `switchdist` to `cutoff`. For a complete description of the non-bonded force parameters see section 4.5.

- `pairlistdist` ⟨ distance between pairs for inclusion in pair lists (Å) ⟩
  Acceptable Values: positive decimal ≥ `cutoff`
  Default Value: `cutoff`+1.0
  Description: A pair list is generated each cycle, containing pairs of atoms for which electrostatics and van der Waals interactions will be calculated. This parameter is used when `switching` is set to `on` to specify the allowable distance between atoms for inclusion in the pair list. This parameter is equivalent to the X-PLOR parameter `CUTNb`. If no atom moves more than one-half of `pairlistdist`−`cutoff` during one cycle, then there will be no jump in electrostatic or van der Waals energies when the next pair list is built. If `pairlistdist` is not specified, the value of `cutoff`+1.0 is used. A value of at least one greater than the maximum of `eleccutoff`, `vdwcutoff`, and `cutoff` is recommended.

- `margin` ⟨ extra length in patch dimension (Å) ⟩
  Acceptable Values: positive decimal
  Default Value: 1.0
  Description: An internal tuning parameter used in determining the size of the cubes of space with which NAMD uses to partition the system. The value of this parameter will not change the physical results of the simulation. For more details about this parameter see the NAMD *Programming Guide*. Unless you are very motivated to get the *very* best possible performance, just leave this value at the default.

- `plMarginCheck` ⟨ perform pairlist margin check? ⟩
  Acceptable Values: `on` or `off`
  Default Value: `on`
  Description: If this parameter is activated, a check will be performed during each timestep to detect atoms that may have violated the margin between the `pairlistdist` and `cutoff` distances. Such violations will be output by NAMD.

Basic dynamics

- `exclude` ⟨ exclusion policy to use ⟩
  Acceptable Values: `none`, `1-2`, `1-3`, `1-4`, or `scaled1-4`
  Description: This parameter specifies which pairs of bonded atoms should be excluded from non-bonded interactions. With the value of `none`, no bonded pairs of atoms will be excluded. With the value of `1-2`, all atom pairs that are directly connected via a linear bond will be excluded. With the value of `1-3`, all `1-2` pairs will be excluded along with all pairs of atoms

that are bonded to a common third atom (i.e., if atom A is bonded to atom B and atom B is bonded to atom C, then the atom pair A-C would be excluded). With the value of `1-4`, all `1-3` pairs will be excluded along with all pairs connected by a set of two bonds (i.e., if atom A is bonded to atom B, and atom B is bonded to atom C, and atom C is bonded to atom D, then the atom pair A-D would be excluded). With the value of `scaled1-4`, all `1-3` pairs are excluded and all pairs that match the `1-4` criteria are modified. The electrostatic interactions for such pairs are modified by the constant factor defined by `1-4scaling`. The van der Waals interactions are modified by using the special 1-4 parameters defined in the parameter files.

- `temperature` ⟨ initial temperature (K) ⟩
  Acceptable Values: positive decimal
  Description: Initial temperature value for the system. Using this option will generate a random velocity distribution for the initial velocities for all the atoms such that the system is at the desired temperature. Either the `temperature` or the `velocities/binvelocities` option must be defined to determine an initial set of velocities. Both options cannot be used together.

- `COMmotion` ⟨ allow center of mass motion? ⟩
  Acceptable Values: `yes` or `no`
  Default Value: `no`
  Description: Specifies whether or not motion of the center of mass of the entire system is allowed. If this option is set to `no`, the initial velocities of the system will be adjusted to remove center of mass motion of the system.

- `dielectric` ⟨ dielectric constant for system ⟩
  Acceptable Values: decimal ≥ 1.0
  Default Value: 1.0
  Description: Dielectric constant for the system. A value of 1.0 implies no modification of the electrostatic interactions. Any larger value will lessen the electrostatic forces acting in the system.

- `1-4scaling` ⟨ scaling factor for 1-4 interactions ⟩
  Acceptable Values: $0 \leq$ decimal ≤ 1
  Default Value: 1.0
  Description: Scaling factor for 1-4 interactions. This factor is only used when the `exclude` parameter is set to `scaled1-4`. In this case, this factor is used to modify the electrostatic interactions between 1-4 atom pairs. If the `exclude` parameter is set to anything but `scaled1-4`, this parameter has no effect regardless of its value.

- `seed` ⟨ random number seed ⟩
  Acceptable Values: positive integer
  Default Value: pseudo-random value based on current UNIX clock time
  Description: The number used to seed the random number generator if `temperature` or `langevin` is selected. This can be used so that consecutive simulations produce the same results. If no value is specified, NAMD will choose a pseudo-random value based on the current UNIX clock time. The random number seed will be output during the simulation startup so that its value is known and can be reused for subsequent simulations. Note that if

Langevin dynamics are used in a parallel simulation (i.e., a simulation using more than one processor) even using the same seed will *not* guarantee reproducible results.

- `rigidBonds` ⟨ controls if and how ShakeH is used ⟩
  Acceptable Values: `none`, `water`, `all`
  Default Value: `none`
  Description: When `rigidBonds` is `all`, the bond between each hydrogen and its mother atom is fixed to the nominal bond length given in the parameter file. When `water` is selected, only the bonds between the hydrogens and the oxygen in water molecules are constrained. For the default case `none`, no lengths are constrained. Only `none` can be used with Langevin dynamics.

- `rigidTolerance` ⟨ allowable bond-length error for ShakeH (Å) ⟩
  Acceptable Values: positive decimal
  Default Value: 0.00001
  Description: The ShakeH algorithm is assumed to have converged when all constrained bonds differ from the nominal bond length by less than this amount.

- `rigidIterations` ⟨ maximum ShakeH iterations ⟩
  Acceptable Values: positive integer
  Default Value: 100
  Description: The maximum number of iterations ShakeH will perform before giving up on constraining the bond lengths. If the bond lengths do not converge, a warning message is printed, and the atoms are left at the final value achieved by ShakeH. Although the default value is 100, convergence is usually reached after fewer than 10 iterations.

DPMTA parameters These parameters control the options to DPMTA, the algorithm used to provide full electrostatic interactions. DPMTA is a modified version of the FMA (Fast Multipole Algorithm) and, unfortunately, most of the parameters still refer to FMA rather than DPMTA for historical reasons. Don't be confused!

For a further description of how exactly full electrostatics are incorporated into NAMD, see section 4.5.4. For a greater level of detail about DPMTA and the specific meaning of its options, see the DPMTA distribution which is available via anonymous FTP from the site `ftp.ee.duke.edu` in the directory `/pub/SciComp/src`.

- `FMA` ⟨ use full electrostatics? ⟩
  Acceptable Values: `on` or `off`
  Default Value: `off`
  Description: Specifies whether or not the DPMTA algorithm from Duke University should be used to compute the full electrostatic interactions. If set to `on`, DPMTA will be used with a multiple timestep integration scheme to provide full electrostatic interactions as detailed in section 4.5.4. Because of the way this feature is currently implemented, PVM must be running for DPMTA to function, even if NAMD is running only on one processor.

- `FMALevels` ⟨ number of levels to use in multipole expansion ⟩
  Acceptable Values: positive integer
  Default Value: 5
  Description: Number of levels to use for the multipole expansion. This parameter is only

used if `FMA` is set to `on`. A value of 4 should be sufficient for systems with less than 10,000 atoms. A value of 5 or greater should be used for larger systems.

- `FMAMp` ⟨ number of multipole terms to use for FMA ⟩
  Acceptable Values: positive integer
  Default Value: 8
  Description: Number of terms to use in the multipole expansion. This parameter is only used if `FMA` is set to `on`. If the `FMAFFT` is set to `on`, then this value must be a multiple of 4. The default value of 8 should be suitable for most applications.

- `FMAFFT` ⟨ use DPMTA FFT enhancement? ⟩
  Acceptable Values: `on` or `off`
  Default Value: `on`
  Description: Specifies whether or not the DPMTA code should use the FFT enhancement feature. This parameter is only used if `FMA` is set to `on`. If `FMAFFT` is set to `on`, the value of `FMAMp` must be set to a multiple of 4. This feature offers substantial benefits only for values of `FMAMp` of 8 or greater. This feature will substantially increase the amount of memory used by DPMTA.

- `FMAtheta` ⟨ DPMTA theta parameter (radians) ⟩
  Acceptable Values: decimal
  Default Value: 0.715
  Description: This parameter specifies the value of the theta parameter used in the DPMTA calculation. The default value is based on recommendations by the developers of the code.

- `FMAFFTBlock` ⟨ blocking factor for FMA FFT ⟩
  Acceptable Values: positive integer
  Default Value: 4
  Description: The blocking factor for the FFT enhancement to DPMTA. This parameter is only used if both `FMA` and `FMAFFT` are set to `on`. The default value of 4 should be suitable for most applications.

- `FullDirect` ⟨ calculate full electrostatics directly? ⟩
  Acceptable Values: `yes` or `no`
  Default Value: `no`
  Description: Specifies whether or not direct computation of full electrostatics should be performed. The direct computation of electrostatics is not intended to be used during real calculations, but rather as a testing or comparison measure. Because of the $\mathcal{O}(N^2)$ computational complexity for performing direct calculations, this is *much* slower than using FMA to compute full electrostatics.

Multiple timestep parameters NAMD allows efficient full electrostatic evaluation through the use of multiple time stepping. The computational aspects of these parameters are discussed in more detail in section 5.2.1.

- `MTSAlgorithm` ⟨ MTS algorithm to be used ⟩
  Acceptable Values: `naive`, `verletI`, `verletX`
  Default Value: `naive`

Description: Specifies the multiple timestep algorithm used to integrate the long range forces. We recommend `verletI` because of its conservation properties, with `stepspercycle` set to 4. If Langevin dynamics is used, then `verletX` might be a better choice. See section 5.2.1 for the differences between these methods.

- `longSplitting` ⟨ how should long and short range forces be split? ⟩
  Acceptable Values: `sharp`, `xplor`, `c1`
  Default Value: `sharp`
  Description: Specifies the manner in which the long and short range forces are combined. We recommend using the `c1` splitting. See section 5.2.1 for more details.

Harmonic constraints The following describes the parameters for the harmonic constraints feature of NAMD. Actually, this feature should be referred to as harmonic restraints rather than constraints, but for historical reasons the terminology of harmonic constraints has been carried over from X-PLOR. This feature allows a harmonic restraining force to be applied to any set of atoms in the simulation. For further details, see section 5.1.8.

- `constraints` ⟨ Are constraints active? ⟩
  Acceptable Values: `on` or `off`
  Default Value: `off`
  Description: Specifies whether or not harmonic constraints are active. If it is set to `off`, then no harmonic constraints are computed. If it is set to `on`, then harmonic constraints are calculated using the values specified by the parameters `consref`, `conskfile`, `conskcol`, and `consexp`.

- `consexp` ⟨ exponent for harmonic constraint energy function ⟩
  Acceptable Values: positive, even integer
  Default Value: 2
  Description: Exponent to be used in the harmonic constraint energy function. This value must be a positive integer, and only even values really make sense. This parameter is used only if `constraints` is set to `on`.

- `consref` ⟨ PDB file containing constraint reference positions ⟩
  Acceptable Values: UNIX file name
  Default Value: `coordinates`
  Description: PDB file to use for reference positions for harmonic constraints. Each atom that has an active constraint will be constrained to the position specified in this file. If no value is given and constraints are active, then the same PDB file specified by `coordinates` will be used instead, constraining atoms to their initial positions.

- `conskfile` ⟨ PDB file containing force constant values ⟩
  Acceptable Values: UNIX filename
  Default Value: `coordinates`
  Description: PDB file to use for force constants for harmonic constraints. If this parameter is not specified, then the PDB file containing initial coordinates specified by `coordinates` is used.

- `conskcol` ⟨ column of PDB file containing force constant ⟩
  Acceptable Values: `X`, `Y`, `Z`, `O`, or `B`

Default Value: `0`
Description: Column of the PDB file to use for the harmonic constraint force constant. This parameter may specify any of the floating point fields of the PDB file, either X, Y, Z, occupancy, or beta-coupling (temperature-coupling). Regardless of which column is used, a value of 0 indicates that the atom should not be constrained. Otherwise, the value specified is used as the force constant for that atom's restraining potential (in kcal/mol/Å^2). **NOTE:** NAMD actually calculates the constraints potential with $U = k(x - x_0)^d$ and the force with $F = -dk(x-x_0)^{d-1}$, where d is the exponent `consexp`. This means that for the force constant k given in the PDB file, the actual force constant is dk in calculations.

Moving constraints The following describes the parameters for the moving harmonic constraint feature of NAMD. This feature allows a harmonic restraining force to be applied to any *one* atom in the simulation and the restraining reference position to be moved.

- `movingConstraints` ⟨ Activate moving constraints? ⟩
  Acceptable Values: `on` or `off`
  Default Value: `off`
  Description: Turns moving constraints `on` and `off`. If set to `on`, then `movingConsVel` and `movingConsAtom` must be defined.

- `movingConsVel` ⟨ velocity of the reference position movement (Å/timestep) ⟩
  Acceptable Values: vector
  Description: The velocity of the reference position movement. Gives both absolute value and direction.

- `movingConsAtom` ⟨ index (0-based) of the constrained atom ⟩
  Acceptable Values: non-negative integer
  Description: The index of the atom constrained to the moving reference position. If the atom is numbered i in the PDB file, $i - 1$ must be specified. NAMD will crash if the atom is not constrained in the `conskfile`.

Energy minimization parameters As described in section 5.2.2, NAMD has the capability of performing energy minimization using a simple quenching scheme. While this algorithm is not the most rapidly convergent, it is sufficient for most applications. There are only two parameters for minimization: one to activate minimization and another to specify the maximum movement of any atom.

- `minimization` ⟨ Perform energy minimization? ⟩
  Acceptable Values: `on` or `off`
  Default Value: `off`
  Description: Turns energy minimization `on` or `off`.

- `maximumMove` ⟨ maximum distance an atom can move during each step (Å) ⟩
  Acceptable Values: positive decimal
  Default Value: $0.75 \times$ `cutoff/stepsPerCycle`
  Description: Maximum distance that an atom can move during any single timestep of minimization. This is to insure that atoms do not go flying off into space during the first few timesteps when the largest energy conflicts are resolved.

Langevin dynamics parameters As described in section 5.2.3, NAMD is capable of performing Langevin dynamics, where additional damping and random forces are introduced to the system. This capability is based on that implemented in X-PLOR which is detailed in the X-PLOR *User's Manual* [6].

- `langevin` ⟨ use Langevin dynamics? ⟩
  Acceptable Values: `on` or `off`
  Default Value: `off`
  Description: Specifies whether or not Langevin dynamics active. If set to `on`, then the parameter `langevinTemp` must be set and the parameters `langevinFile` and `langevinCol` can optionally be set to control the behavior of this feature. Also, `rigidBonds` must be set to none.

- `langevinTemp` ⟨ temperature for Langevin calculations (K) ⟩
  Acceptable Values: positive decimal
  Description: Temperature to which atoms affected by Langevin dynamics will be adjusted. This temperature will be roughly maintained across the affected atoms through the addition of friction and random forces.

- `langevinFile` ⟨ PDB file containing Langevin parameters ⟩
  Acceptable Values: UNIX filename
  Default Value: `coordinates`
  Description: PDB file to use for the Langevin coupling coefficients for each atom. If this parameter is not specified, then the PDB file specified by `coordinates` is used.

- `langevinCol` ⟨ column of PDB from which to read coefficients ⟩
  Acceptable Values: `X`, `Y`, `Z`, `O`, or `B`
  Default Value: `O`
  Description: Column of the PDB file to use for the Langevin coupling coefficients for each atom. The coefficients can be read from any floating point column of the PDB file. A value of 0 indicates that the atom will remain unaffected.

Temperature coupling parameters NAMD is capable of performing temperature coupling, in which forces are added or reduced to simulate the coupling of the system to a heat bath of a specified temperature. This capability is based on that implemented in X-PLOR which is detailed in the X-PLOR *User's Manual* [6].

- `tCouple` ⟨ perform temperature coupling? ⟩
  Acceptable Values: `on` or `off`
  Default Value: `off`
  Description: Specifies whether or not temperature coupling is active. If set to `on`, then the parameter `tCoupleTemp` must be set and the parameters `tCoupleFile` and `tCoupleCol` can optionally be set to control the behavior of this feature.

- `tCoupleTemp` ⟨ temperature for heat bath (K) ⟩
  Acceptable Values: positive decimal
  Description: Temperature to which atoms affected by temperature coupling will be adjusted. This temperature will be roughly maintained across the affected atoms through the addition of forces.

- `tCoupleFile` ⟨ PDB file with tCouple parameters ⟩
  Acceptable Values: UNIX filename
  Default Value: `coordinates`
  Description: PDB file to use for the temperature coupling coefficient for each atom. If this parameter is not specified, then the PDB file specified by `coordinates` is used.

- `tCoupleCol` ⟨ column of PDB from which to read coefficients ⟩
  Acceptable Values: `X`, `Y`, `Z`, `O`, or `B`
  Default Value: `O`
  Description: Column of the PDB file to use for the temperature coupling coefficient for each atom. This value can be read from any floating point column of the PDB file. A value of 0 indicates that the atom will remain unaffected.

Temperature rescaling parameters NAMD allows equilibration of a system by means of temperature rescaling. Using this method, all of the velocities in the system are periodically rescaled so that the entire system is set to the desired temperature. The following parameters specify how often and to what temperature this rescaling is performed.

- `rescaleFreq` ⟨ number of timesteps between temperature rescaling ⟩
  Acceptable Values: positive integer
  Description: The equilibration feature of NAMD is activated by specifying the number of timesteps between each temperature rescaling. If this value is given, then the `rescaleTemp` parameter must also be given to specify the target temperature.

- `rescaleTemp` ⟨ temperature for equilibration (K) ⟩
  Acceptable Values: positive decimal
  Description: The temperature to which all velocities will be rescaled every `rescaleFreq` timesteps. This parameter is valid only if `rescaleFreq` has been set.

Spherical harmonic boundary conditions NAMD provides spherical harmonic boundary conditions. These boundary conditions can consist of a single potential or a combination of two potentials as described in section 5.1.10. The following parameters are used to define these boundary conditions.

- `sphericalBC` ⟨ use spherical boundary conditions? ⟩
  Acceptable Values: `on` or `off`
  Default Value: `off`
  Description: Specifies whether or not spherical boundary conditions are to be applied to the system. If set to `on`, then `sphericalBCr1` and `sphericalBCk1` must be defined, and `sphericalBCexp1`, `sphericalBCr2`, `sphericalBCk2`, and `sphericalBCexp2` can optionally be defined.

- `sphericalBCr1` ⟨ radius for first boundary condition (Å) ⟩
  Acceptable Values: positive decimal
  Description: Distance at which the first potential of the boundary conditions takes effect. This distance is a radius from the center of mass.

- `sphericalBCk1` ⟨ force constant for first potential ⟩
  Acceptable Values: non-zero decimal
  Description: Force constant for the first harmonic potential. A positive value will push atoms toward the center of mass, and a negative value will pull atoms away from the center of mass.

- `sphericalBCexp1` ⟨ exponent for first potential ⟩
  Acceptable Values: positive, even integer
  Default Value: 2
  Description: Exponent for first boundary potential, most likely either 2 or 4.

- `sphericalBCr2` ⟨ radius for second boundary condition (Å) ⟩
  Acceptable Values: positive decimal
  Description: Distance at which the second potential of the boundary conditions takes effect. This distance is a radius from the center of mass. If this parameter is defined, then `spericalBCk2` must also be defined.

- `sphericalBCk2` ⟨ force constant for second potential ⟩
  Acceptable Values: non-zero decimal
  Description: Force constant for the second harmonic potential. A positive value will push atoms toward the center of mass, and a negative value will pull atoms away from the center of mass.

- `sphericalBCexp2` ⟨ exponent for second potential ⟩
  Acceptable Values: positive, even integer
  Default Value: 2
  Description: Exponent for second boundary potential, most likely either 2 or 4.

Applied electric field NAMD provides the ability to apply a constant electric field to the molecular system being simulated. There are two parameters that control this feature.

- `eFieldOn` ⟨ apply electric field? ⟩
  Acceptable Values: `yes` or `no`
  Default Value: `no`
  Description: Specifies whether or not an electric field is applied.

- `eField` ⟨ electric field vector ⟩
  Acceptable Values: vector of decimals
  Description: Vector which describes the electric field to be applied.

4.3 Translation between NAMD and X-PLOR configuration parameters

NAMD was designed to provide many of the same molecular dynamics functions that X-PLOR provides. As such, there are many similarities between the types of parameters that must be passed to both X-PLOR and NAMD. This section describes relations between similar NAMD and X-PLOR parameters.

- **NAMD Parameter:** `cutoff, eleccutoff, vdwcutoff`
  X-PLOR Parameter: `CTOFNB`
  When full electrostatics are not in use within NAMD, these parameters have exactly the same meaning — the distance at which electrostatic and van der Waals forces are truncated. When full electrostatics are in use within NAMD, the meaning is still very similar. The van der Waals force is still truncated at the specified distance, and the electrostatic force is still computed at every timestep for interactions within the specified distance. However, the NAMD integration uses multiple time stepping to compute electrostatic force interactions beyond this distance every `stepspercycle` timesteps.

- **NAMD Parameter:** `vdwswitchdist`
  X-PLOR Parameter: `CTONNB`
  Distance at which the van der Waals switching function becomes active.

- **NAMD Parameter:** `pairlistdist`
  X-PLOR Parameter: `CUTNb`
  Distance within which interaction pairs will be included in pairlist.

- **NAMD Parameter:** `1-4scaling`
  X-PLOR Parameter: `E14Fac`
  Scaling factor for 1-4 pair electrostatic interactions.

- **NAMD Parameter:** `dielectric`
  X-PLOR Parameter: `EPS`
  Dielectric constant.

- **NAMD Parameter:** `exclude`
  X-PLOR Parameter: `NBXMod`
  Both parameters specify which atom pairs to exclude from non-bonded interactions. The ability to ignore explicit exclusions is *not* present within NAMD, thus only positive values of `NBXMod` have NAMD equivalents. These equivalences are

  - `NBXMod=1` is equivalent to `exclude=none` — no atom pairs excluded,
  - `NBXMod=2` is equivalent to `exclude=1-2` — only 1-2 pairs excluded,
  - `NBXMod=3` is equivalent to `exclude=1-3` — 1-2 and 1-3 pairs excluded,
  - `NBXMod=4` is equivalent to `exclude=1-4` — 1-2, 1-3, and 1-4 pairs excluded,
  - `NBXMod`=5 is equivalent to `exclude=scaled1-4` — 1-2 and 1-3 pairs excluded, 1-4 pairs modified.

- **NAMD Parameter:** `switching`
  X-PLOR Parameter: `SHIFt, VSWItch, and TRUNcation`
  Activating the NAMD option `switching` is equivalent to using the X-PLOR options `SHIFt` and `VSWItch`. Deactivating `switching` is equivalent to using the X-PLOR option `TRUNcation`.

- **NAMD Parameter:** `temperature`
  X-PLOR Parameter: `FIRSttemp`
  Initial temperature for the system.

- **NAMD Parameter:** `rescaleFreq`
  X-PLOR Parameter: `IEQFrq`
  Number of timesteps between velocity rescaling.
- **NAMD Parameter:** `rescaleTemp`
  X-PLOR Parameter: `FINAltemp`
  Temperature to which velocities are rescaled.
- **NAMD Parameter:** `restartname`
  X-PLOR Parameter: `SAVE`
  Filename prefix for the restart files.
- **NAMD Parameter:** `restartfreq`
  X-PLOR Parameter: `ISVFrq`
  Number of timesteps between the generation of restart files.
- **NAMD Parameter:** `DCDfile`
  X-PLOR Parameter: `TRAJectory`
  Filename for the position trajectory file.
- **NAMD Parameter:** `DCDfreq`
  X-PLOR Parameter: `NSAVC`
  Number of timesteps between writing coordinates to the trajectory file.
- **NAMD Parameter:** `velDCDfile`
  X-PLOR Parameter: `VELOcity`
  Filename for the velocity trajectory file.
- **NAMD Parameter:** `velDCDfreq`
  X-PLOR Parameter: `NSAVV`
  Number of timesteps between writing velocities to the trajectory file.
- **NAMD Parameter:** `numsteps`
  X-PLOR Parameter: `NSTEp`
  Number of simulation timesteps to perform.

4.4 Sample configuration files

This section contains some simple example NAMD configuration files to serve as templates.

This file shows a simple configuration file for alanin. It performs basic dynamics with no output files or special features.

```
timestep        0.5
numsteps        10000

cwd             /scratch               #  Specify a working directory

structure       alanin.psf
parameters      alanin.params
coordinates     alanin.pdb

exclude         scaled1-4
1-4scaling      0.4
outputname      output
margin          1.0
stepspercycle   10
temperature     300.0

switching       on
switchdist      7.0
cutoff          8.0
pairlistdist    9.0
```

This file is again for alanin, but shows a slightly more complicated configuration. A coordinate trajectory file and a set of restart files are produced every ten timesteps. Langevin dynamics are also active in this configuration.

```
timestep        0.5

numsteps        10000

structure       alanin.psf
parameters      alanin.params
coordinates     alanin.pdb
exclude         scaled1-4
1-4scaling      0.4
outputname      output
margin          1.0
stepspercycle   10
temperature     300.0

switching       on
switchdist      7.0
cutoff          8.0
pairlistdist    9.0

DCDfile         alanin.dcd
DCDfreq         10

restartname     alanin.restart
restartfreq     10

langevin        on
langevinTemp    300.0
langevinCol     O
```

This file shows another simple configuration file for alanin, but this time with full electrostatics using DPMTA.

```
timestep        0.5
numsteps        10000

cwd             /scratch        #  Specify a working directory

structure       alanin.psf
parameters      alanin.params
coordinates     alanin.pdb

exclude         scaled1-4
1-4scaling      0.4
outputname      output
margin          1.0
stepspercycle   10
temperature     300.0

switching       on
switchdist      7.0
cutoff          8.0
pairlistdist    9.0

# DPMTA parameters
FMA             on
FMAMp           8
FMALevels       3
FMAFFT          on
FMAFFTBlock     4
```

4.5 Non-bonded interaction parameters and computations

NAMD has a number of options that control the way that non-bonded interactions are calculated. These options are interrelated and can be quite confusing, so this section attempts to explain the behavior of the non-bonded interactions and how to use these parameters.

4.5.1 Non-bonded van der Waals interactions

The simplest non-bonded interaction is the van der Waals interaction. In NAMD, van der Waals interactions are always truncated at the cutoff distance, specified by `vdwcutoff` or `cutoff`. The main option that effects van der Waals interactions is the `switching` parameter. With this option set to `on`, a smooth switching function will be used to truncate the van der Waals potential energy smoothly at the cutoff distance. A graph of the van der Waals potential with this switching function is shown in Figure 1. If `switching` is set to `off`, the van der Waals energy is just abruptly truncated at the cutoff distance, so that energy may not be conserved.

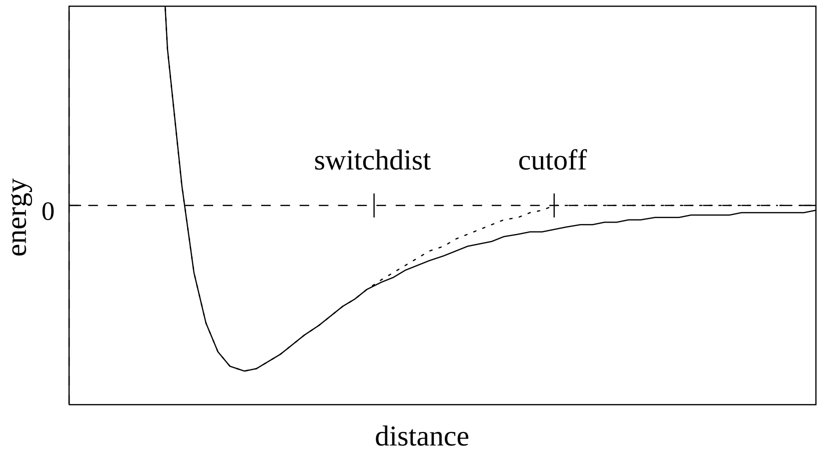

Figure 1: Graph of van der Waals potential with and without the application of the switching function. With the switching function active, the potential is smoothly reduced to 0 at the cutoff distance. Without the switching function, there is a discontinuity where the potential is truncated.

See section 5.1.4 for more details. The switching function used is based on the X-PLOR switching function. The parameter `switchdist` specifies the distance at which the switching function should start taking effect to bring the van der Waals potential to 0 smoothly at the cutoff distance. Thus, the value of `switchdist` must always be less than that of `cutoff`.

4.5.2 Non-bonded electrostatic interactions

The handling of electrostatics is slightly more complicated due to the incorporation of DPMTA to provide full electrostatic interactions. There are two cases to consider, one where DPMTA is used to provide full electrostatics and the other where electrostatics are truncated at a given distance.

First let us consider the latter case, where electrostatics are truncated at the cutoff distance. Using this scheme, all electrostatic interactions beyond a specified distance are ignored, that is, assumed to be zero. If `switching` is set to `on`, rather than having a discontinuity in the potential at the cutoff distance, a shifting function is applied to the electrostatic potential as shown in Figure 2. As this figure shows, the shifting function shifts the entire potential curve so that the curve intersects the x-axis at the cutoff distance. This shifting function is based on the shifting function used by X-PLOR. See section 5.1.4 for more details.

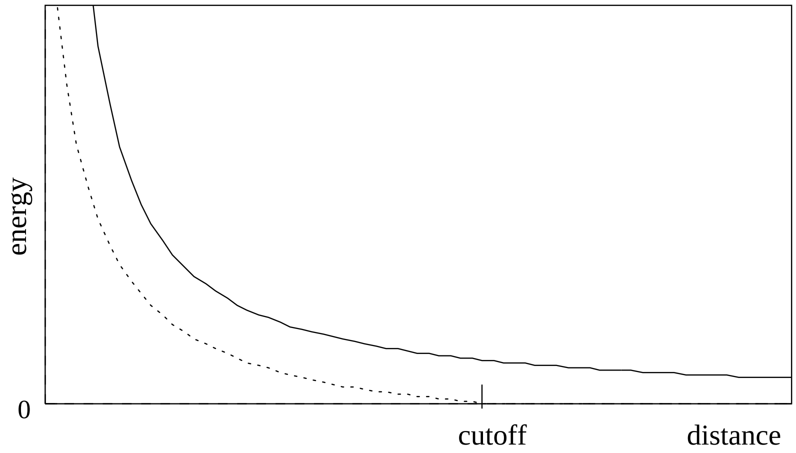

Figure 2: Graph showing an electrostatic potential with and without the application of the shifting function.

Next, consider the case where full electrostatics are calculated. In this case, the electrostatic interactions are not truncated at any distance. In this scheme, the `cutoff` parameter has a slightly different meaning for the electrostatic interactions — it represents the *local interaction distance*, or distance within which electrostatic pairs will be directly calculated every timestep. Outside of this distance, interactions will be calculated only periodically using DPMTA. These forces will be applied using a multiple timestep integration scheme as described in section 4.5.4.

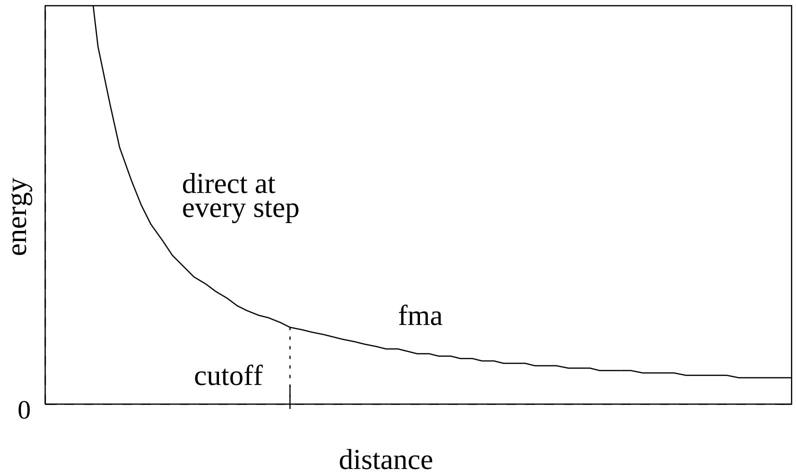

Figure 3: Graph showing an electrostatic potential when full electrostatics are used within NAMD, with one curve portion calculated directly and the other calculated using DPMTA.

4.5.3 Pair list generation

The last critical parameter for non-bonded interaction calculations is the parameter `pairlistdist`. To reduce the cost of performing the non-bonded interactions, NAMD uses a *non-bonded pair list.* This list contains all pairs of atoms for which non-bonded interactions should be calculated. Performing the search for pairs of atoms that should have their interactions calculated is an expensive operation. Thus, the pair list is only calculated periodically, once per cycle. Unfortunately, pairs of atoms move relative to each other during the steps between preparation of the pair list. Because of this, if the pair list were built to include only those pairs of atoms that are within the cutoff distance when the list is generated, it would be possible for atoms to drift closer together than the cutoff distance during subsequent timesteps and yet not have their non-bonded interactions calculated.

Let us consider a concrete example to better understand this. Assume that the pairlist is built once every ten timesteps and that the cutoff distance is 8.0 Å. Consider a pair of atoms A and B that are 8.1 Å apart when the pairlist is built. If the pair list includes only those atoms within the cutoff distance, this pair would not be included in the list. Now assume that after five timesteps, atoms A and B have moved to only 7.9 Å apart. A and B are now within the cutoff distance of each other, and should have their non-bonded interactions calculated. However, because the non-bonded interactions are based solely on the pair list and the pair list will not be rebuilt for another five timesteps, this pair will be ignored for five timesteps causing energy not to be conserved within the system.

To avoid this problem, the parameter `pairlistdist` allows the user to specify a distance greater than the `cutoff` distance for pairs to be included in the pair list, as shown in Figure 4. Pairs that are included in the pair list but are outside the cutoff distance are simply ignored. So in the above example, if the `pairlistdist` were set to 10.0 Å, then the atom pair A and B would be included in the pair list, even though the pair would initially be ignored because they are further apart than the cutoff distance. As the pair moved closer and entered the cutoff distance, because the pair was already in the pair list, the non-bonded interactions would immediately be calculated and energy conservation would be preserved. The value of `pairlistdist` should be chosen such that no atom pair moves more than `pairlistdist` − `cutoff` in one cycle. This will insure energy conservation.

4.5.4 Full electrostatic integration

To further reduce the cost of computing full electrostatics, NAMD uses a multiple time stepping integration scheme. In this scheme, the total force acting on each atom is broken into two pieces, a local component and a long range component. The local force component is defined in terms of a *local interaction distance.* The local force component consists of all bonded interactions as well as all non-bonded interactions for pairs that are separated by less than the local interaction distance. The long range component consists only of electrostatic interactions outside of the local interaction distance. Since the long range forces are slowly varying, they are not evaluated every timestep. Instead, they are evaluated every k timesteps, where each set of k timesteps is referred to as a *cycle.* The value of k is specified by the NAMD parameter `stepspercycle`. There are several ways in which the long range forces may be combined with the short range forces. See section 5.2.1 for an explanation of the various multiple time stepping options available. For appropriate values of k, it is believed that the error introduced by this infrequent evaluation of the long range forces is modest compared to the error already incurred by the use of the numerical (Verlet) integrator. The performance of NAMD with the use of DPMTA to provide full electrostatics approximately

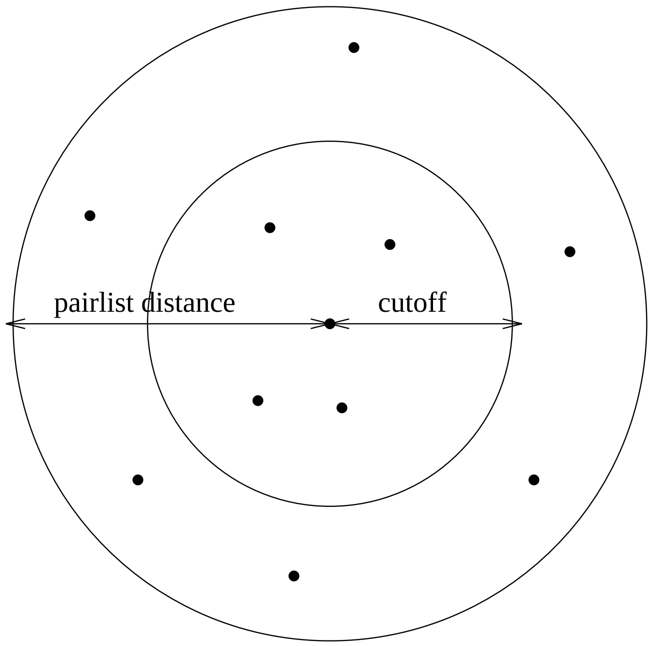

Figure 4: Depiction of the difference between the cutoff distance and the pair list distance. The pair list distance specifies a sphere that is slightly larger than that of the cutoff so that pairs are allowed to move in and out of the cutoff distance without causing energy conservation to be disturbed.

doubles the run time of a simulation as compared to NAMD with an 8.0 Å electrostatic cutoff. Improved methods for incorporating these long range forces are currently being investigated, with the intention of improving accuracy as well as reducing the frequency of long range force evaluations.

In the scheme described above, the van der Waals forces are still truncated at the local interaction distance. Thus, the van der Waals cutoff distance forms a lower limit to the local interaction distance. While this is believed to be sufficient, there are investigations underway to remove this limitation and provide full van der Waals calculations in $\mathcal{O}(N)$ time as well.

5 Computational Details

This section details exactly what NAMD calculates during a simulation. Section 5.1 describes the various components of the force field that are used. Section 5.2 describes the integration choices that are used. Section 5.3 lists the units, constants, and conversion factors used.

5.1 Force fields

The force fields used by NAMD are compatible with those used by the programs X-PLOR and CHARMm. It includes energy terms for electrostatics, van der Waals, linear bonds, angular bonds, dihedral bonds, improper bonds, and Hydrogen bonds. The potential energy function used by NAMD is

$$E_{\mathrm{total}} = E_{\mathrm{elec}} + E_{\mathrm{vdw}} + E_{\mathrm{bond}} + E_{\mathrm{angle}} + E_{\mathrm{d/i}} + E_{\mathrm{constraint}}$$

The following subsections describe the individual energy terms in detail as well as how to determine the forces associated with each type of interaction.

5.1.1 Electrostatics

Electrostatics describes the force resulting from the interaction between two charged particles. The electrostatic energy between two atoms i and j is described by Coulomb's Law as

$$E_{\mathrm{elec}} = \epsilon_{14} \frac{C q_i q_j}{\epsilon_0 |\vec{r}_{ij}|}$$

where

ϵ_{14} = Scaling factor for 1-4 interactions. This is zero for 1-2 and 1-3 interactions and is equal to 1.0 for any interaction other than a modified 1-4 interaction. The value used for modified 1-4 interactions is specified in the configuration file using the `1-4scaling` parameter. (See section 4.)

$C = 2.31 \times 10^{-19}$ J nm

q_i = charge of atom i from the PSF file

q_j = charge of atom j from the PSF file

ϵ_0 = Dielectric constant specified in the configuration file using the parameter `dielectric`. (See section 4.)

$\vec{r}_{ij} = \vec{r}_j - \vec{r}_i$, the vector from atom i to atom j

$|\vec{r}_{ij}|$ = length of $\vec{r}_{ij}$, which is equal to the calculated distance between atoms i and j

5.1.2 Van der Waals

The van der Waals interactions describe the forces resulting from local interactions of atoms. The van der Waals energy between two atoms i and j is described by

$$E_{\mathrm{vdw}} = \frac{A}{|\vec{r}_{ij}|^{12}} - \frac{B}{|\vec{r}_{ij}|^{6}}$$

where

A = constant determined as described below

B = constant determined as described below

$|\vec{r}_{ij}|$ = calculated distance between atoms i and j

The constants A and B can be specified for a pair of atom types explicitly in the parameter file using the `NBFix` command. With this command, values of A and B for normal interactions and modified 1-4 interactions are specified explicitly.

If the `NBFix` command is not used, the constants A and B are calculated using the parameters σ_{ij} and ϵ_{ij} using the equations

$$A = 4\sigma_{ij}^{12}\epsilon_{ij}$$
$$B = 4\sigma_{ij}^{6}\epsilon_{ij}$$

where σ_{ij} and ϵ_{ij} are calcuated from the σ and ϵ values specifed for the atom types of atoms i and j in the `NBOnd` commands in the parameter file using the equations

$$\sigma_{ij} = \frac{\sigma_{ii} + \sigma_{jj}}{2}$$
$$\epsilon_{ij} = \sqrt{\epsilon_{ii}\epsilon_{jj}}$$

where

σ_{ii} = σ value for atom i

ϵ_{ii} = ϵ value for atom i

Also, σ and ϵ for a pair can be related to the minimum energy, E_{min}, and minimum distance, R_{min} by the equations

$$R_{\mathrm{min}} = \sigma\sqrt[6]{2}$$
$$E_{\mathrm{min}} = -\epsilon$$

For modified 1-4 interactions, the σ_{ij} and ϵ_{ij} values are calculated as above, except that the σ_{ii}, ϵ_{ii}, σ_{jj}, and ϵ_{jj} are replaced by the σ_{ii}^{14}, ϵ_{ii}^{14}, σ_{jj}^{14}, and ϵ_{jj}^{14} that are specified by the `NBOnd` lines of the parameter files. Section 5.1.3 describes when the modified parameters are used.

5.1.3 Bonded exclusions

Certain pairs of atoms are excluded from electrostatic and van der Waals calculations because of their bonded interactions. The rules to apply in choosing bonded exclusions are specifed in the configuration file using the `exclude` parameter. The choices for exclusions are `none`, `1-2`, `1-3`, `1-4`, and `scaled1-4`. With `none`, no atom pairs are excluded. With `1-2`, only atoms that are connected via a linear bond are excluded. With `1-3`, any pair of atoms connected via a bond or bonded to a common third atom are excluded. With `1-4`, any atoms that are connected by a linear bond, or a sequence of two bonds, or a sequence of three bonds, are excluded. With `scaled1-4`, exclusions are applied in the same manner as the `1-3` setting, but those pairs that are connected by a sequence of 3 bonds are calculated using the modified 1-4 methods described rather than the standard force calculations.

5.1.4 Switching functions

NAMD allows the cutting off of the electrostatic and van der Waals forces to be handled in two ways. The default way is to simply truncate the forces at the cutoff value specified in the configuration file using the parameter `cutoff`. In this case, if the distance r between two atoms is less than the cutoff, then the forces are calculated as specified above. If r is greater than the cutoff, then the forces and energies are simply set to 0. But this method leads to a discontinuity in the force field. As atoms move within the cutoff distance, there electrostatic and van der Waals energies suddenly jump from 0 to some finite value.

The other means of dealing with these cutoffs provided by NAMD is switching functions. These functions smoothly bring the forces and energies to 0 at the cutoff distance to avoid any discontinuities in the force field. These switching functions are equivalent to those used in X-PLOR when the `VSWitch` and `SHIft` options are specified. They are activated using the `switching` parameter in the configuration file.

There are different switching functions used for electrostatics and van der Waals. For electrostatics, the energy function is modified to be

$$E_{\text{elec}} = \epsilon_{14} \frac{C q_i q_j}{\epsilon_0 |\vec{r}_{ij}|} S_{\text{elec}}(|\vec{r}_{ij}|)$$

with the switching function defined by

$$S_{\text{elec}}(|\vec{r}_{ij}|) = \begin{cases} \left(1 - \frac{|\vec{r}_{ij}|^2}{{R_{\text{off}}}^2}\right)^2 & \text{if } |\vec{r}_{ij}| < R_{\text{off}}, \\ 0 & \text{if } R_{\text{off}} \leq |\vec{r}_{ij}|. \end{cases}$$

For van der Waals interactions, the energy function is modified to be

$$E_{\text{vdw}} = \left(\frac{A}{|\vec{r}_{ij}|^{12}} - \frac{B}{|\vec{r}_{ij}|^{6}} \right) S_{\text{vdw}}(|\vec{r}_{ij}|)$$

with the switching function defined by

$$S_{\text{vdw}}(|\vec{r}_{ij}|) = \begin{cases} 1 & \text{if } |\vec{r}_{ij}| \leq R_{\text{on}}, \\ \frac{\left(R_{\text{off}}^2 - |\vec{r}_{ij}|^2\right)^2 \left(R_{\text{off}}^2 + 2|\vec{r}_{ij}|^2 - 3R_{\text{on}}^2\right)}{\left(R_{\text{off}}^2 - R_{\text{on}}^2\right)^3} & \text{if } R_{\text{on}} < |\vec{r}_{ij}| < R_{\text{on}}, \\ 0 & \text{if } R_{\text{off}} \leq |\vec{r}_{ij}|. \end{cases}$$

where

R_{on} is specified by the configuration value `switchdist`

R_{off} is specified by the configuration value `cutoff`

5.1.5 Bonds

Bonds describe a linear bond between two atoms. These bonds are described by simple harmonic springs. The energy of a bond between atoms i and j is given by

$$E_{\text{bond}} = k(|\vec{r}_{ij}| - r_0)^2$$

where

k = spring constant specified in the parameter file for this bond type

$|\vec{r}_{ij}|$ = calculated distance between atoms i and j

r_0 = rest distance of the bond specified in the parameter file for this bond type

5.1.6 Angles

Angles describe angular bonds between three atoms. These bonds are modeled as harmonic angular springs. The energy of such a bond between atoms i, j, and k is given by

$$E_{\text{angle}} = k_\theta(\theta - \theta_0)^2 + k_{\text{ub}}(|\vec{r}_{ik}| - r_{\text{ub}})^2$$

where

k_θ = force constant specified in the parameter file for this bond type

θ = calculated angle between the vector that connect atoms i and j and the vector that connects atoms k and j. This angle is calculated using the formula:

$$\theta = \cos^{-1}\left(\frac{\vec{r}_{ij} \cdot \vec{r}_{kj}}{|\vec{r}_{ij}||\vec{r}_{kj}|}\right)$$

θ_0 = rest angle for this bond specified in the parameter file for this angle type

k_{ub} = Urey-Bradley constant, which defaults to zero

r_{ub} = rest distance for the Urey-Bradley term

5.1.7 Dihedrals and impropers

Dihedral and improper bonds model the interaction between 4 bonded atoms. They are modeled by an angular spring between the planes formed by the first 3 atoms and the second 3 atoms. The energy for a dihedral or improper between atoms i, j, k, and l is given by

$$E_{\mathrm{d/i}} = k(1 + \cos(n\phi + \delta))$$

if $n > 0$ or

$$E_{\mathrm{d/i}} = k(\phi - \delta)^2$$

if $n = 0$, where

k = force constant specified in the parameter file for this bond type

ϕ = calculated angle between the plane formed by atoms i, j, and k and the plane formed by atoms j, k, and l

n = periodicity of the bond specified in the parameter file for this bond type

δ = phase shift specified in the parameter file for this bond type

5.1.8 Harmonic constraints

Harmonic constraints provide a mechanism for holding certain parts of a molecule relatively immobile during a simulation. It allows specific atoms to be held to a reference position. The energy for an atom i that is constrained is

$$E_{\mathrm{constraint}} = k_i(|\vec{r}_i - \vec{r}_{\mathrm{ref}}|)^e$$

where

k_i = force constant defined for atom i

$\vec{r}_i$ = current position of atom i

$\vec{r}_{\mathrm{ref}}$ = reference position atom i is constrained to

e = exponent defined for all constraints

The reference positions, force constaints, exponent, and atoms which are constrained are defined by the user using several configuration options. For more details, see the description of the options `constraints`, `consexp`, `conskfile`, `conskcol`, and `consref` in section 4.

NOTE: NAMD actually calculates the constraints potential with $E = k(x - x_0)^d$ and the force with $F = -dk(x - x_0)^{d-1}$, where d is the exponent `consexp`. The result is that a force constant k specified in the PDB file is effectively the force constant dk in calculations (usually $d = 2$).

5.1.9 Moving constraints

Moving constraints allow NAMD to run steered molecular dynamics (SMD) simulations. This feature is based on the harmonic constraints feature, and effectively allows for the reference position of a specified atom to move with a constant velocity $\vec{v}$. The potential for this atom is computed according to

$$E_{\rm smd}(t) \;=\; k_i \;(|\vec{r}_{\rm ref}(t) \;-\; \vec{r}_i(t)|)^e$$

where

t = time

$\vec{r}_i$ = position of atom i

$\vec{r}_{\rm ref}(t) = \vec{r}(0) + \vec{v}t$, current reference position (position of the restraint point)

e = exponent defined for all constraints

In addition to the usual harmonic constraint parameters, the constant velocity vector $\vec{v}$ must be specified by parameter `movingConsVel`, and the index of the constrained atom (using 0-based indexing) must be specified by parameter `movingConsAtom`. See section 4.

The way the moving constraints work is that the moving reference position is calculated every integration time step with $\vec{v}$ in Å/timestep and t the current timestep (i.e., `firstTimestep` plus however many timesteps have passed since the beginning of NAMD run). Therefore, one should be careful when restarting simulations to appropriately update either the `firstTimestep` parameter in the NAMD configuration file or the reference position specified in the reference PDB file.

5.1.10 Spherical boundary conditions

NAMD provides spherical boundary conditions that consist of a single or two superimposed potential functions. The potential energy for these energies is given by

$$E_{\rm sphere} = k_{\rm sphere}(|\vec{r}_i - \vec{r}_{\rm center}| - r_{\rm sphere})^{exp_{\rm sphere}}$$

if $|\vec{r}_i - \vec{r}_{\rm center}| > r_0$ and $E_{\rm sphere} = 0$ otherwise, where

$k_{\rm sphere}$ = force constant

$\vec{r}_i$ = current position of atom i

$\vec{r}_{\rm center}$ = center of the spherical boundary

$r_{\rm sphere}$ = radius of the spherical boundary

$exp_{\rm sphere}$ = exponent for the potential function

The force applied by this potential is given by

$$\vec{F}_{\rm sphere} = \left(exp_{\rm sphere} k_{\rm sphere}\,(|\vec{r}_i - \vec{r}_{\rm center}| - r_{\rm sphere})^{exp_{\rm sphere}-1}\right)\hat{r}_{i,\rm center}$$

where

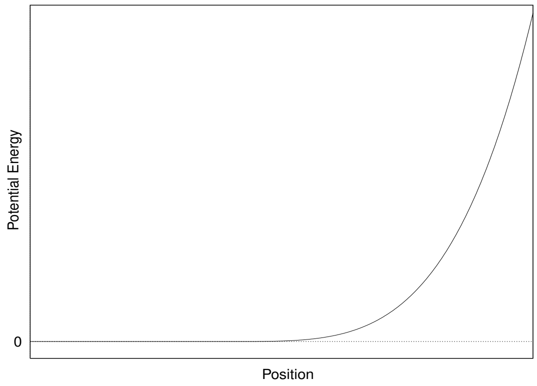

Figure 5: Potential energy function for a spherical boundary condition using a single potential.

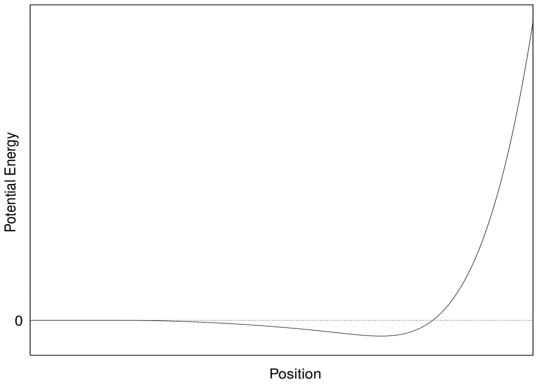

Figure 6: Potential energy function for a spherical boundary condition using two potentials, one a negative quadratic and another a positive quartic, to approximate surface tension.

$$\hat{r}_{i,\text{center}} = \text{the unit vector in the direction from atom } i \text{ to the center of the sphere.}$$

Thus a positive force constant will cause a force that moves atoms back in to the center of the sphere and a negative force constant will force atoms away from the sphere.

For simple harmonic boundary conditions, a single potential with an exponent of 2 or 4 may be used to obtain a potential like that shown in figure 5. It may be desirable, however, to simulate a surface tension affect, where a small harmonic well around the sphere boundary is desired. To obtain such an effect, two potentials of the form shown above can be superimposed. By combining a potential with a negative force constant and an exponent of two with a potential with a positive force constant and a exponent of four and a slightly larger raidus, a potential such as that shown in figure 6 can be obtained.

Currently, the spheres must be centered around the initial center of mass of the system. The force constant, radius, and exponents for each potential are defined in the configuration file using the options `sphericalBCk1`, `sphericalBCr1`, `sphericalBCexp1`, `sphericalBCk2`, `sphericalBCr2`, and `sphericalBCexp2`. These parameters are described in section 4.

5.2 Integration

NAMD provides several ways to adjust the integration for the force fields described in the previous section. These integration options are discussed within the following sections.

5.2.1 Verlet method

NAMD uses the velocity form of the Verlet (leapfrog) method for integration. Beginning at a timestep n and given the position, velocity, and force acting on each atom, X_n, V_n, and F_n, the following equations are used to obtain values for the next step.

$$
\begin{aligned}
V_{n+\frac{1}{2}} &= V_n + \frac{\Delta t}{2} M^{-1} F_n \\
X_{n+1} &= X_n + \Delta t V_{n+\frac{1}{2}} \\
F_{n+1} &= F(X_{n+1}) \\
V_{n+1} &= V_{n+\frac{1}{2}} + \frac{\Delta t}{2} M^{-1} F_{n+1}
\end{aligned}
$$

There are slightly different formulations of the Verlet algorithm that rely on positions X_n and X_{n-1}, but it is felt that such formulations suffer from a higher degree of round-off error than the velocity formulation.

Multiple time stepping can be used along with FMA (DPMTA) to make full electrostatics cheaper to compute. (Computing full electrostatics directly using the `FullDirect` option is meant only for very small systems or testing purposes.) Full electrostatic simulations are enabled by setting the `FMA` parameter `on`. The long range electrostatic forces are evaluated only at the cycle boundaries, that is to say, every `stepspercycle` steps. The parameter `MTSAlgorithm` determines the multiple time stepping method to be used:

- `naive` — The long range force for each step is determined using constant extrapolation from the most recent cycle boundary step.
- `verletI` — The long range force scaled by `stepspercycle` is added as an impulse only on cycle boundary steps.
- `verletX` — The long range force for each step is determined using linear extrapolation from the previous two cycle boundary steps.

See [3] for more details about the Verlet I and Verlet X methods.

The long and short range electrostatic forces may be smoothly combined using a *splitting function* defined by the `longSplitting` parameter:

- `sharp` — No smoothing is done. The forces are simply added together.
- `xplor` — The X-PLOR switching function for the van der Waals potential is applied to the electrostatic potential. See section 5.1.4 for details.
- `c1` — Each electrostatic force interaction is split into

$$
\begin{aligned}
\vec{F}_{\text{elec}}^{\text{short}} &= S(|\vec{r}_{ij}|)\vec{F}_{\text{elec}} \\
\vec{F}_{\text{elec}}^{\text{long}} &= [1 - S(|\vec{r}_{ij}|)]\vec{F}_{\text{elec}}
\end{aligned}
$$

by the continuously differentiable scalar function

$$S(|\vec{r}_{ij}|) = \begin{cases} 1 & \text{if } |\vec{r}_{ij}| \le R_{\text{on}}, \\ 2\left(\frac{|\vec{r}_{ij}|-R_{\text{on}}}{R_{\text{off}}-R_{\text{on}}}\right)^3 - 3\left(\frac{|\vec{r}_{ij}|-R_{\text{on}}}{R_{\text{off}}-R_{\text{on}}}\right)^2 + 1 & \text{if } R_{\text{on}} < |\vec{r}_{ij}| \le R_{\text{off}}, \\ 0 & \text{if } R_{\text{off}} < |\vec{r}_{ij}|. \end{cases}$$

We recommend the use of the `c1` splitting [7].

5.2.2 Minimization

NAMD provides a simple form of a depth first search to perform energy minimization. This method uses a modified form of the Verlet method described above. It is modified in two ways. The first is that the velocity of the particles is set to zero after each timestep. This means that direction of movement of each particle during an integration step will always be in the direction of the gradient, thus createing a depth first serach. The second modification is to place a bound on the movement of an atom during any timestep, since a structure with very high potential energies may experience very rapid movement of atoms during the beginning of minimization. The amount of movement per timestep can be specified using the configuration option `maximumMove`. If no value is specified, NAMD will assume a default value of

$$\frac{0.75 \times \texttt{pairlistDistr}}{\texttt{stepsPerCycle}},$$

where the parameter `pairlistDist` is the bounding distance between pairs for inclusion in pairlists, and the parameter `stepsPerCycle` indicates the number of time steps between pairlist generation. See section 4 for more details about these configuration parameters. The default value for `maximumMove` insures that no atom can move more than 3/4 of the pairlist distance during a cycle. If greater accuracy in the energy calculation is desired, then this value should be reduced.

5.2.3 Langevin dynamics

Simple Langevin dynamics are also provided by NAMD. This consists of adding a random force and subtracting a friction force from each atom during each integration step. The random force is calculated such that the average force is zero and the standard deviation is

$$2K_{\text{b}}T_0 b_i m_i \Delta t$$

where

K_{b} = Boltzman's constant

T_0 = target temperature specified using the configuration parameter `langTemp`

b_i = friction coefficient for atom i

m_i = mass of atom i

Δt = timestep size

The friction force applied is

$$m_i b_i \frac{dx_i}{dt}$$

where

m_i = mass of atom i

b_i = friction coefficient for atom i

In order to apply these forces, NAMD uses the same second-order finite difference approximation as X-PLOR. This approximation uses

$$x_i^{n+1} = \left(1 + \frac{b_i \Delta t}{2}\right)\left(2x_i^n - x_i^{n-1} + F_i^n \frac{\Delta t^2}{m_i} + x_i^{n-1}\frac{b_i \Delta t}{2}\right)$$

to update the position of particle i from step n to step $n+1$ and

$$v_n = \frac{x_{n+1} - x_{n-1}}{2\Delta t}$$

for the velocity at step n.

5.2.4 Velocity rescaling

NAMD allows the user to equilibrate a system to a specified temperature by periodically rescaling the velocities of all particles such that a specified temperature is achieved. This rescaling is achieved by first applying the integration scheme to obtain a set of velocities for timestep n. These velocities are then rescaled to the desired temperature by multiplying by the factor

$$\sqrt{\frac{T_0}{T_{\text{ave}}}}$$

where

T_0 = the desired temperature specified using the configuration option `rescaleTemp`

T_{ave} = the average temperature since the last velocity rescaling

The positions for timestep $n+1$ are then obtained by using these new velocities.

Temperature rescaling may be used with either the Verlet integrator or Langevin dynamics. The number of timesteps between rescalings is specified using the configuration parameter `rescaleFreq`.

5.2.5 Rigid bonds

Rigid water is an option (rigid bond angle and bond lengths). Also bonds between hydrogens and heavy atoms can be constrained to their nominal length during integration. Three options for selecting what bonds to constrain are given in NAMD. The configuration parameter `rigidBonds` can be set to `none`, which indicates no constrained lengths, `water` to constrain only bonds in water molecules, or `all` to constrain all bonds between hydrogens and heavy atoms. Bonds between heavy atoms cannot currently be constrained, and NAMD will not work correctly (even with rigid bonds inactive) if the system contains hydrogen molecules, since no heavy (or *parent*) atom can be found for either hydrogen. The rigid bond code also cannot handle methane molecules, or any other molecule with four or more hydrogens bonded to a heavy atom.

The rigid bonds module sorts the atoms of interest into one of four data structures, according to how many hydrogen atoms are bonded to the parent atom. After applying the normal integration procedure, the atom positions are adjusted to bring the bond lengths back to their neutral length (ie. the length such that the bond exerts no force on either atom). These adjustments are applied in such a way that the center of mass of the bonded atoms does not move. For the one-hydrogen case, this is a straightforward calculation, but for the two- or three-hydrogen cases, an iterative procedure is followed, where each hydrogen-parent atom pair is adjusted until all the hydrogen bonds are within a certain tolerance of the desired length, or a maximum number of iterations is exceeded. These values are set by the `rigidTolerance` and `rigidIterations` parameters. A modified algorithm is applied to waters. Although waters have only two hydrogens, a constraint is also applied to the distance between the two hydrogens and the three-bond procedure followed, so that the angle between the hydrogens is also constrained.

After correcting the atom positions, the calculated velocities must also be adjusted to account for the changed positions. Without velocity correction, the kinetic energy is computed incorrectly. Velocity correction is performed by the Rattle module. The velocities calculated by the normal integration are adjusted by removing the component parallel to the bond, without changing the velocity of the center of mass of the bonded pair.

Constraints: $\chi^\alpha(X) = 0, \alpha = 1, 2, \ldots,$whatever

Algorithm: given $X_n, V_{n-\frac{1}{2}}$

crunch numbers: $F_n =$ forces

half-kick: $\bar{V}_n = V_{n-\frac{1}{2}} + \frac{1}{2}\Delta t M^{-1} F_n$

half-kick: $V_{n+\frac{1}{2}} = \bar{V}_n + \frac{1}{2}\Delta t M^{-1} F_n$

drift: $\bar{X}_{n+1} = X_n + \Delta t V_{n+\frac{1}{2}}$

SHAKE: $X_{n+1} = \bar{X}_{n+1} + \frac{1}{2}\Delta t^2 M^{-1} \sum_\beta \lambda_\beta \chi_x^\beta(X_n)$

where $\{\lambda_\beta\}$ solve

$\{\chi_\alpha(X_{n+1}) = 0\}$

$\{V_{n+\frac{1}{2}} = (X_{n+1} - X_n)/\Delta t\}$

RATTLE: Compute the new velocities using: $V_n = (I - M^{-1}\chi_x^{\mathrm{t}}(\chi_x M^{-1}\chi_x^{\mathrm{t}})^{-1}\chi_x)\bar{V}_n$

Note that the optional RATTLE step is for the purpose of computing kinetic energy. It forces velocities to satisfy velocity constraints, but these adjustments are exactly undone by the next SHAKE.

Rigid bond calculation can be performed with Verlet integration, with or without minimization or velocity rescaling. Although NAMD will run with Langevin integration and rigid bonds both active, the equilibrium temperature of the system does not converge to the correct value. Different Langevin equations are needed to produce the correct velocity distribution, which have not yet been added to NAMD.

5.3 Units, constants, and conversion factors

This section defines the units, constant values, and conversion factors that are used by NAMD.

5.3.1 Units

The units currently used by NAMD are:

Measure	Unit	Symbol
length	angstroms	Å
time	picoseconds	ps
energy	kilocalories per mole	kcal/mol
mass	atomic mass unit	amu
charge	electron charge	e
temperature	degrees kelvin	K

5.3.2 Constants

The constant values used by NAMD are:

Symbol	Value	Name
K_{b}	1.987191×10^{-3} kcal/(mol·K)	Boltzman's constant
N_{a}	6.022045×10^{23} 1/mol	Avogadro's number
C	332.0636 kcal·Å/(mol·e^2)	Coulomb's constant
e	1.6021892 coulombs	electron charge
π	3.1415926535898	pi

5.3.3 Conversion factors

The conversion factors currently used by NAMD are:

$$1 \text{ kcal} = 4184.0 \text{ joules}$$

$$1 \text{ amu} = 1.6605655 \times 10^{-27} \text{ kilograms}$$

$$1 \text{ Å} = 1.0 \times 10^{-10} \text{ meters}$$

References

[1] M. P. Allen and D. J. Tildesley. *Computer Simulation of Liquids.* Oxford University Press, New York, 1987.

[2] F. C. Bernstein, T. F. Koetzle, G. J. B. Williams, J. E. F. Meyer, M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi, and M. Tasumi. The protein data bank: A computer-based archival file for macromolecular structures. *J. Mol. Biol.*, 112:535–542, 1977.

[3] J. J. Biesiadecki and R. D. Skeel. Dangers of multiple-time-step methods. *J. Comput. Phys.*, 109(2):318–328, 1993.

[4] J. Board, Z. Hakura, W. Elliot, and W. Rankin. Scalable variants of multipole-accelerated algorithms for molecular dynamics applications. Technical Report TR94-006, Duke University, Dept. of Elec. Engr., 1994.

[5] B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan, and M. Karplus. Charmm: A program for macromolecular energy, minimization, and dynamics calculations. *J. Comp. Chem.*, 4(2):187–217, 1983.

[6] A. T. Brünger. *X-PLOR, Version 3.1, A System for X-ray Crystallography and NMR.* The Howard Hughes Medical Institute and Department of Molecular Biophysics and Biochemistry, Yale University, 1992.

[7] D. D. Humphreys, R. A. Friesner, and B. J. Berne. A multiple-time-step molecular dynamics algorithm for macromolecules. *J. Phys. Chem.*, 98:6885–6892, 1994.

[8] J. A. McCammon and S. C. Harvey. *Dynamics of Proteins and Nucleic Acids.* Cambridge University Press, Cambridge, 1987.