Manual 01 for MD++

Introduction to MD++

Keonwook Kang and Wei Cai

January 10, 2007

1 Overview

MD++ is a molecular dynamics simulation package written in C++. It is originally developed by Wei Cai when he was a graduate student at MIT and still being updated to include more features. MD++ is supposed to run in a Unix/Linux environment. However, if you have cygwin[1], a linux-like environment installed in your Windows machine, you can also enjoy MD++ there. Currently, this code is designed mainly for studying the atomic behavior in solid crystal materials. It can also be extended to study other systems, such as fluids, polymers and bio-molecules. This document is the first of a series of manuals written for those new to MD simulations. These manuals explain how to use MD++ with simple examples.

2 Installation

The latest version of MD++ code can be downloaded from the web site [1], `http://micro.stanford.edu/`. Here we assume that the reader is familiar with basic Unix/Linux shell commands.[2] Before you download MD++ code, we recommend that that you make a sub-directory `Codes` in your home directory,

```
$ mkdir Codes
$ cd Codes
```

Save the MD++ package in this directory (`~/Codes`). Suppose the downloaded file is named `md++-2007-01-07.tar.gz`, then we can extract it by,

```
$ tar -zxvf md++-2007-01-07.tar.gz
```

[1] For download and installation, visit http://www.cygwin.com/. You must download and install the full package of cygwin. See http://micro.stanford.edu/~caiwei/Forum/2004-12-12-MD++/cygwin.html for installation instructions.

[2] For those who are not familiar with the Unix/Linux environment, we suggest the book "A practical guide to Linux commands, editors and shell programming" by Mark G. Sobell. You can also easily find several good web sites that list all the shell commands.

This command creates the `MD++` sub-directory and extracts all the files into it. Go to the `MD++` directory and make a `runs` directory if it does not exist.

```
$ cd MD++
$ mkdir runs
```

To compile the codes in `R`elease mode[3] in a Linux system, type

```
$ make all build=R
```

In a different system (such as Windows), you can specify the `SYS` variable in the above command line, such as,

```
$ make all build=R SYS=cygwin
```

provided that cygwin is installed. You can also compile MD++ code in an Apple/Machintosh computer, using

```
$ make all SYS=mac build=R
```

By default `SYS=gpp`, which corresponds to the GNU c++ compiler. Use `SYS=intel` to activate the intel c/c++ compiler `icc`, which produces a faster executable than `gpp`.

After compilation, you have executable files such as `fs_gpp`, `eam_gpp`, and `sw_gpp` in the `bin` directory.[4] If you just want to compile one executable file, such as `fs_gpp`, use

```
$ make fs build=R
```

The name of each execution file stands for the potential model used in the MD simulation. For example, `fs` stands for the Finnis-Sinclair (FS)potential[2], `lj` for the Lennard-Jones (LJ) pair potential[3], `eam` for the embedded-atom method (EAM) potential[4], and `sw` for the Stillinger-Weber (SW) potential[5, 6]. Each potential model is applicable to a set of atomic species. For example, the FS potential was designed for body-centered-cubic (BCC) metals such as Mo, Ta, and W. The SW potential was designed for diamond-cubic semiconductors such as Si and Ge. The executable `md_gpp` does not contain any potential model. Hence it cannot be used to compute any material properties but it can still be used to display atomic structures.
If you modify the source code (in the `src` directory), you need to recompile

[3] If `build=R`, MD++ is compiled with the `-O3` option. By default `build=D`, corresponding to the debug mode, in which MD++ is compiled with the `-g` option. If `build=R` the executable files run faster but the compilation time is slightly longer than if `build=D`.

[4] If `SYS=cygwin`, the executable files would be `fs_cygwin.exe`, `eam_cygwin.exe`, etc.

MD++. We recommend you to clean up the previous executables and rebuild by,

```
$ make clean; make all build=R
```

To see more help on `make`, type

```
$ make help
```

To check if the installation and compilation is successful, try to run a script in the `Examples` directory, e.g.

```
$ bin/fs_gpp scripts/Examples/example01-mo.script
```

If you want to stop an MD++ run, press `Ctrl-c` or close the visualization window.

3 MD++ Directories

After installation, the `MD++` directory will contain the following sub-directories.

```
$ ls
bin        Doc        makefile   potentials   scripts   Tools
CSD-book   Examples   matlab     README       src
```

The contents in each directory is described below.

`src`	source codes
`bin`	Binary executable files generated by `make`
`Doc`	Manuals
`scripts`	Input script files for running MD++
`runs`	Where the simulation outputs are stored
`potentials`	Data files to specify various potential models
`Examples`	Some example input script files
`Tools`	External tools such as `atomeye` for visualization
`matlab`	Matlab codes for postprocessing

4 Script File

To execute a molecular simuation, the MD++ code reads a script file which contains a series of commands and variable assignments for the simulation. To create or modify a script file in a Unix/Linux environment, you can use *vi*, *emacs*, or *nedit* editor, or any of your favorite editors. For example,

```
$ vi scripts/example01-mo.script
```

or

```
$ emacs scripts/example01-mo.script &
```

or

```
$ nedit scripts/example01-mo.script &
```

You can see how a script file looks like by opening example script file. All script files have a common head region,

```
# -*-shell-script-*-
setnolog                # No log file saved
setoverwrite            # Overwrite a log file
dirname = runs/mo-example       # Specify output directory
```

The first line tells the *Emacs* editor to use the shell-script mode to display the file. In a script file, you can comment out a line or part of the line by the pound sign, `#`. You can use # to leave your own comment or explanation anywhere in the script. The `setnolog` command tells MD++ not to create a log file. If `setnolog` is commented out, then all outputs to the screen will be redirected to a file `A.log` output directory specified by `dirname`, which also contains all the other output files.
The simulation stops whenever it reaches the `quit` command. Sometimes it is useful to put `sleep` before the `quit` command, such as

```
sleep quit     # System gets sleep for 600 sec before quit
```

The `sleep` command keeps the visualization window alive for a while to give us a chance to look at the atomic structure. Other commands and variables in the script file will be explained in more detail later.

5 Visualization Window Control

When you run MD++, a new window usually pops up through which you can observe the motion of atoms. We will refer to this window as the "visualization window". There are several short keys to manipulate the object displayed in this window. For example, to translate the object press 't' and then use arrow keys. To rotate it, use 'r' and arrow keys. To see more help, press F1, and the following will be printed to the screen (if `setnolog` is commented out)

```
Mouse drag to rotate
Hot Keys:
F1        : display this message
Up        : rotate up
Down      : rotate down
```

```
Left      : rotate left
Right     : rotate right
PgUp      : rotate counterclockwise
PgDn      : rotate clockwise
Home      : back to initial viewpoint
Space     : stop rotate
p         : toggle pause
t         : translation
s         : scaling
d         : move projection infinity point
r         : rotation
f         : toggle pbc enableness
m         : toggle drawframe
g         : pbc glide
x         : pbc shift in x
y         : pbc shift in y
z         : pbc shift in z
w         : print window specification
F9        : output gif
F10       : output postscript
```

References

[1] This is the companion website for the book by V. V. Bulatov and W. Cai, *Computer Simulations of Dislocations*, Oxford University Press (2006).

[2] M. W. Finnis and J. E. Sinclair, *Philos. Mag. A.* **50** 45 (1984).

[3] J. E. Lennard-Jones, "Cohesion" *Proceedings of the Physical Society* **43** 461-482 (1931).

[4] M. S. Daw and M. I. Baskes, "Embedded-atom method: Derivation and application to impurities, surfaces, and other defects in metals", *Phys. Rev. B* **29** 6443 (1984).

[5] F. H. Stillinger and T. A. Weber, "Computer simulation of local order in condensed phases of silicon", *Phys. Rev. B* **31** 5262 (1985).

[6] H. Balamane, T. Halicioglu, and W. A. Tiller, "Comparative study of silicon empirical interatomic potentials", *Phys. Rev. B* **46** 2250 (1992).