Tight Binding Molecular Dynamics

-

User Manual & Code Documentation

Developers: Siroki, Gleb; Najafi, Behnam; Hasan, Hikmatyar; Cimbaro, Luca; Blumenthal, Lars; Ramos Fernandez, Eduardo.

March 27, 2015

Imperial College London

-

EPSRC Centre for Doctoral Training

on

Theory and Simulation of Materials

Group Programming Project

1 Installation

1.1 Personal Systems

To install the package on a personal (Linux-based) system, follow the following steps:

1. Unarchive "`TBMD.zip`" into a directory of your choice. This creates a directory called "`/TBMD/`".
2. Navigate to "`/TBMD/src/`" from the command line tool and execute the following command:

   ```
   make
   ```

This will create the required object files in the "`/TBMD/build/`" directory. You can then follow the steps in Section 2 to run the program. Note that if any changes are made to the code, step 2 above has to be repeated in order to create new object files. Testing was performed on Mac OS X and Ubuntu 14.0 (64-bit) operating systems; we cannot guarantee compatibility with other operating systems.

1.2 High Performance Computing Clusters

To install the package on a high-performance computing cluster, follow the following steps:

1. Unarchive "`TBMD.zip`" into your work directory. This creates a directory called "`/TBMD/`".
2. Navigate to "`/TBMD/"` from the command line tool and execute the following command:

   ```
   ./build.sh
   ```

This will create the required object files in the "`/TBMD/build/`" directory. You can then follow the steps in Section 2 to run the program. Note that if any changes are made to the code, step 2 above has to be repeated in order to create new object files. Testing was performed on cx1 clusters in Imperial College London. For other HPC systems, the compatibility of this software is highly dependent on the architecture, hardware and configuration of the processors.

2 Running

2.1 Personal Systems

1. Navigate to "`/TBMD/build/`" from the command line tool and execute the following command:

   ```
   ./tbmd -c input -i structure.cell -o output.dat
   ```

   - "`./tbmd`" specifies the object file to be executed. This should not be changed.
   - "`input`" specifies the name of the input file. The contents of this file can be changed by the user to specify the required parameters (see section 3).
   - "`structure.cell`" specifies the name of the file that contains the initial structure of the Carbon atoms.
   - "`output.dat`" specifies the name of the output file that stores the parameters during the evolution of the structure.

2. A movie file: "`movie.xyz`" is created for visualisation purposes. This file contains the coordinates of the atoms at user-specified intervals.

3. The final values of the normal modes of the structure along with the associated eigenvectors are written to the file: "`NormalModes-C#`", where "`#`" is the number of atoms.

2.2 High Performance Computing Clusters

1. Open the file "`/TBMD/run.sh`" using a text editor such as VIM. In the sixth line of "`/TBMD/run.sh`", replace the letter the letter "`X`" with the path of the directory containing "`/TBMD/`". This line originally reads:

   ```
   X/TBMD/build/tbmd -c X/TBMD/input -i X/TBMD/structure.cell -o X/TBMD/output.dat
   ```

   - "`X/TBMD/build/tbmd`" specifies the path to the object file to be executed.
   - "`X/TBMD/input`" specifies the path to the input file. The contents of this file can be changed by the user to specify the required parameters (see section 3).
   - "`X/TBMD/structure.cell`" specifies the path to the file that contains the initial structure of the Carbon atoms.
   - "`X/TBMD/output.dat`" specifies the path to the output file that stores the parameters during the evolution of the structure.

2. Make sure that "`/TBMD/input`" and "`/TBMD/structure.cell`" exist.

3. While in the "`/TBMD/`" directory, execute the following command:

   ```
   qsub run.sh
   ```

4. A movie file: "`movie.xyz`" is created for visualisation purposes. This file contains the coordinates of the atoms at user-specified intervals.

5. The final values of the normal modes of the structure along with the associated eigenvectors are written to the file: "`NormalModes-C#`", where "`#`" is the number of atoms.

3 Input Files

3.1 File: `input`

The `input` file has a format based on key-value pairs. It consists of a number of assignments (tags) of the form:

TAGNAME = VALUE

Each tag appears in a new line and each "VALUE" can be changed by the user to intialise the variable "TAGNAME". Empty lines and comment lines (starting with %) in the `input` file are ignored. These tags, along with the allowed "VALUE" that they take are explained below. The order that they appear in the `input` file does not matter.

Specifies the time step used in the MD simulation in femto-seconds. It accepts any double-precision floating-point number. The default initialisation is:

TIME_STEP = 0.1

Specifies the number of steps in the MD simulation. Any integer value is accepted and the default initialisation is:

NUMBER_OF_STEPS = 30000

Specifies the type of thermostat that is used in the MD simulation. It only accepts one of the (self-explanatory) values: LANGEVIN, ANDERSON, OVRVO. The default initialisation is:

THERMOSTAT = LANGEVIN

Sets the target temperature of the system. It accepts any double-precision floating-point number. The default initialisation is:

TEMPERATURE = 273.00

Specifies the the damping factor in the MD simulation. It accepts any double-precision floating-point number. The default initialisation is:

MD_GAMMA = 0.05

Specifies the level of detail of the output file (file name specified on execution). The options are: BASIC, DETAILED, LAST_STEP, DEBUG. For details on the structure of these options, see Section 4. The default initialisation is:

OUTPUT_FILE_MODE = DETAILED

Specifies the resolution of the output file. It accepts any integer value and limits the number of recorded time-steps in the output file to NUMBER_OF_STEPS / OUTPUT_DATA_RATE. The default initialisation is:

OUTPUT_DATA_RATE = 10

Specifies whether a file containing details about the normal modes is created. The allowed values are: true, false. The default initialisation is:

NORMAL_MODES = true

Specifies whether a movie file is created. The allowed values are: true, false. The default initialisation is:

CREATE_MOVIE_FILE = true

Specifies the frame rate of the movie file. It accepts any integer value and limits the number of recorded frames in the movie file to NUMBER_OF_STEPS / MOVIE_FRAME_RATE. The default initialisation is:

MOVIE_FRAME_RATE = 10

3.2 File: `structure.cell`

This file specifies the initial coordinates of the Carbon atoms. The first line specifies the number of atoms, followed by an empty line. Following this, each line begins with the letter C, followed by a single space, and the position of the atom in Cartesian coordinates (x,y,z), separated by a single space. For example, the `structure.cell` file for a chain of 3 Carbon atoms, lined up uniformly on the x-axis, with a pair separation of 2.0Å looks like:

3

C 0.0 0.0 0.0
C 2.0 0.0 0.0
C 4.0 0.0 0.0

4 Output Files

4.1 File: `output.dat`

Regardless of the mode, the basis structure of the output file comprises of rows and columns. Each row corresponds to values of variables at a single time-step of the MD simulation. The time elapsed between each row is given by the product of TIME_STEP and OUTPUT_DATA_RATE, as specified in the input file. Each column represents a variable such as total energy of the system and each column is separated by a single space. The variables stored as columns depend on the value of the tag OUTPUT_FILE_MODE, as specified in the input file.

The column headings in the case of OUTPUT_FILE_MODE = BASIC are:

Time	Total Energy	Temperature

The column headings in the case of OUTPUT_FILE_MODE = DETAILED are:

Time	Kinetic Energy	Bond Energy	Repulsive Energy	Total Energy	Temperature

The column headings in the case of OUTPUT_FILE_MODE = DEBUG are:

Time	XXXXXX	XXXXXX

In the case of OUTPUT_FILE_MODE = LAST_STEP, the same columns as DETAILED output appear, but only values at the final time-step are printed.

4.2 File: `movie.xyz`

This file is used for visualisation purposes and its format is optimised for reading using a molecular viewer package such as "Jmol" [1]. The initial coordinates of the atoms is first stored to the file in the same way as the `structure.cell` file, followed by an empty line. This block of coordinates corresponds to the initial frame of the `movie.xyz` file. The time elapsed between each block of stored coordinates is given by the product of TIME_STEP and MOVIE_FRAME_RATE.

4.3 File: `NormalModes-C#`

After an MD simulation where the system has been relaxed, the eigenvalues and eigenvectors of the dynamical matrix of the system are computed, and these eigenvalues corresponds to the normal modes. The file `NormalModes-C#`, where `#` is the number of atoms is created and stores information about the normal modes and the corresponding eigenvectors. This can be exported to an external package such as MATLAB for further analysis.

The first column stores all of the normal mode values; for a system of N atoms, there will be 3N normal modes and therefore `NormalModes-C#` consists of 3N rows. Following each normal mode, a TAB character is inserted and the corresponding 3N-dimensional eigenvector is stored as a row-vector, with its elements also separated by a TAB character.

References

[1] Jmol: an open-source Java viewer for chemical structures in 3D. http://www.jmol.org/