omegagene Documentation

Release 0.38

myPresto/omegagene team

Jun 27, 2016

Author Kota Kasahara

CHAPTER

ONE

DOCUMENTATION STRUCTURE

The documention consists of the following three components:

1.1 Omegagene Users' Documentation

Contents:

1.1.1 System Preparation

Author Kota Kasahara

Input files

The input files required for MD simulations by *omegagene* are compatible with myPresto/psygene. *omegagene* requires

1. Restart file (including the initial coordinates and velocities)
2. Topology file (generated by Tplgene program)
3. Shake file (generated by SHAKEinp program)

In addition, two configuration files are required.

4. System configuration file (system.cfg)
5. Simulation configuration file (md.cfg)

omega_toolkit generates a binary file by combining some of the input files.

```
python2.7 ${OMEGATK}/mdinput_generator.py -i system.cfg -o system.cls
```

Example of system.cfg file is shown below:

system.cfg:

```
--fn-i-tpl          et1.tpl     ; Topology file
--fn-i-initial-pdb  et1.pdb     ; Structure file in .pdb format
--fn-i-restart      et1.restart ; Restart file
--cell-x            61.2425     ; Specifying the each length of cell-axes
--cell-y            61.2425
--cell-z            61.2425
--fn-i-shake        system.shk  ; SHAKE file
;--fn-i-ttp-v-mcmd-inp      ttp_v_mcmd.inp  ; For V-McMD
;--fn-i-ttp-v-mcmd-initial  start.vert      ; For V-McMD
```

md_input.cfg:

```
--mode                  md                  ; Only the keyword "md" is accepted
--integrator            leapfrog-presto     ; "leapfrog-presto" or "zhang"
--thermostat            scaling     ; Thermostat algorithm
                                    ; "none" or "scaling"
--cutoff                12.0        ; Cutoff distance in angestrome
--n-steps               10          ; The number of steps to be calculated
--time-step             2.0         ; Integration time step in fs
--electrostatic         zero-dipole ; Only "zero-dipole" is accepted for GPU
                                    ; "zero-quadrupole", "zero-octupole", and
                                    ; "zero-hexadecapole" are also aceppted for CPU.
--ele-alpha             0           ; Dumping factor for ZD
                                    ;   only 0 is OK for GPU version
--temperature           300         ; The target temperature
;--temperature-init     10          ; The initial temperature
;--heating-steps        10000       ; The number of steps for heating
--print-interval-log    1           ; Interval steps for printing logs
--print-interval-coord  1           ; Interval steps for output the trajectory
--fn-o-coord            trpc.trr    ; File name for the output trajectory
--format-o-coord        presto      ; File format for the output trajectory
                                    ; only "presto" is allowed.
;--fn-o-log              et1.log     ; Not used in the current version
;--fn-o-energy           et1.ene     ; Not used in the current version
--nsgrid-cutoff         13.0        ; Neighbor search cutoff radius (angstrom)
--nsgrid-update-intvl   50          ; Interval time steps for update the neighbor search grid

; Expanded ensemble
;--expanded-ensemble             v-mcmd          ; "none", "v-mcmd", or "v-aus"
;--fn-o-vmcmd-log                ttp_v_mcmd.out  ; Output filename
;--fn-o-expand-lambda            mule.ene        ; Output filename
;--print-interval-expand-lambda  1               ; Interval steps for the output
;--format-o-expand-lambda        ascii           ; "ascii", or "binary"
```

Execute

omegagene executes a MD simulation with two configuration files provided as command line arguments. The current version of omegagene output log messages to the standard output. Redirecting to a log file is recommended.

```
omegagene --inp et1.cls --cfg md_input.cfg > log.txt
```

1.1.2 In/Out Files

Author Kota Kasahara

Input Files

1. System configuration file (.cfg)
2. Simulation configuration file (.cfg)
3. Structure file (.pdb)
4. Initial coordinates and velocities file (.restart)
5. Topology file (.tpl)
6. Integrated binary (.cls)
7. SHAKE setting file (.shk)
8. V-McMD (or V-AUS) setting files (.inp, .vert)

9. V-AUS restart file (.dat)
10. Atom group definition file (.inp)
11. Distance restraint file (.inp)

System configuration file (.cfg)

The input file describing configurations about a simulation system. Other some input files are specified in this file, and they are integrated into the integrated binary file (.cls) by using *mdinput_generator.py* program. In the configuration file, a set of a key and value(s) is specified in each line.

- **--fn-i-tpl** md.tpl
  - The file name of the topology file (.tpl).
- **--fn-i-initial-pdb** md.pdb
  - The file name of the structure file (.pdb).
- **--fn-i-restart** md.restart
  - The file name of the initial coordinates and velocities file (.restart)
- **--cell-x** 61.2425
- **--cell-y** 61.2425
- **--cell-z** 61.2425
  - The lengths of the periodic boundary cell in each axis in angestrome unit.
- **--fn-i-shake** system.shk
  - The file name of the shake setting file.
- **--fn-i-ttp-v-mcmd-inp** ttp_v_mcmd.inp
- **--fn-i-ttp-v-mcmd-initial** start.vert
  - The file name of the V-McMD setting files.
- **--fn-i-atom-group** atom_groups.inp
  - The file name of the atom group definition file.
- **--fn-i-dist-restraint** dist_rest.inp
  - The file name of the distance restraint setting file.
- **--fn-i-aus-restart** aus_rest.dat

Simulation configuration file (.cfg)

The input file describing configurations about simulation conditions.

Common configurations

- **--mode** md
  - Only the keyword “md” is valid.
- **--gpu-device-id** 0
  - The device ID of GPGPU board to be used.
- **--integrator** leapfrog-presto
  - Type of integrator

  - leapfrog-presto * The leap frog algorithm. * For NVE, NVT (rescaling), SHAKE
  - zhang * The integrator by Zhang [Zhang_1997] * For the Hoover-Evans thermostat ("hoover-evans") * SHAKE cannot be applied.
- **--thermostat** scaling
  - none * NVE
  - scaling * Scaling velocities.
  - hoover-evans * Hoover-Evans. This can be applied for the integrator "zhang".
- **--cutoff** 12.0
  - The cutoff length for non-bonded potential (angestrome unit)
- **--n-steps** 10
  - The number of steps of the simulation.
- **--time-step** 2.0
  - The integration time step
- **--electrostatic** zero-dipole
  - "zero-dipole" * The Zero-dipole summation (ZD) method developed by Fukuda [Fukuda_2011] .
  - "zero-quadrupole"
  - "zero-octupole"
  - "zero-hexadecapole" * The Zero-multipole summation (ZM) method developed by Fukuda [Fukuda_2014] .
  - For GPU mode, only zero-dipole with –ele-alpha 0.0 is acceptable.
- **--ele-alpha** 0.0
  - The dumping factor fo ZM method.
  - For GPU mode, only 0.0 is acceptable
- **--temperature** 300
  - Temperature for the thermostat.
- –temperature-init * The initial temperature. Default value is the temperature specified by "–temperature" setting. * With this setting, "–heating-steps" should be set.
- –heating-steps * The temperature is linearly increased or decreased from the –temperature-init to –temperature during the steps specified in this setting.
- **--print-interval-log** 1
  - Output interval for the log (the standard output)
- **--print-interval-coord** 1
  - Output interval for the trajectory file.
- **--fn-o-coord** et1.trr
  - Output file name for the trajectory file.
- **--format-o-coord** gromacs
  - The file format of the trajectory.
  - presto
  - gromacs * (!) The little endian
- **--fn-o-restart** md.restart

  - Output restart file name.
- **--nsgrid-cutoff** 13.0
  - The cutoff length for the neighbor search (angestrome unit).
- **--nsgrid-update-intvl** 10
  - The interval steps for execution of the neighbor search.
- **--com-motion** cancel
  - Settings for canceling the center of mass
  - none
  - cancel * Translation of the center of mass for some specified groups are cancelled. * The groups should be specified in "–com-cancel-group-name"
- **--com-cancel-group-name** grpA
  - The name of an atom group COM motions of which to be cancelled.
  - Multiple values can be specified.

Configuration for restraints

- –dist-restraint harmonic * Functions for distance restraints * none * harmonic
- –dist-restraint-weight * Scaling coefficient for the distance restraints

Configuration for the extended ensemble methods

- **--extended-ensemble** v-mcmd
  - none * Extended ensemble is not used
  - v-mcmd * The TTP-V-McMD method [Higo et al. (2013)]_
  - v-aus * The TTP-V-AUS method [Higo et al. (2015)]_
- **--fn-o-vmcmd-log** ttp_v_mcmd.out
  - Output file name of a virtual-system trajecotry.
- **--fn-o-extended-lambda** mule.ene
  - Output file name of a log of the lambda value.
- **--print-interval-extended-lambda** 1
  - Output interval for the log of the lambda value.
- **--format-o-extended-lambda** ascii
  - File format of the log of the lambda value.
  - ascii
  - binary

#* –aus-type 3 # * Type of AUS reaction coordinate. # * 3 # * The reaction coordinate is defined as the distance between centers of mass of the groups. * –enhance-sigma

- A parameter for the recovery force in V-AUS simulation.
- A margin of the lambda value.
- –enhance-recovery-coef

Initial coordinates and velocityies file (.restart)

This file is compatible for myPresto/Psygene restart file. For the first run, this file with random velocities can be generated by *presto_generate_velocities.py* scripts in the toolkit. At the end of a simulation, final coordinates and velocities will be output at the file specified by *–fn-o-restart* option.

Topology file (.tpl)

This file is compatible for myPresto/Psygene topology file. It can be prepared by using myPresto/TopolgeneX program.

- –fn-i-tpl md.tpl

SHAKE setting file.

This file is compatible for myPresto/Psygene SHAKE file. It can be prepared by using SHAKEinp program.

It should be specified in the system configuration file:

- –fn-i-shake system.shk

V-McMD (or V-AUS) setting files (.inp, .vert)

These fiels are compatible for myPresto/Psygene files.

They should be specified in the system configuration file:

- **--fn-i-ttp-v-mcmd-inp** ttp_v_mcmd.inp
- **--fn-i-ttp-v-mcmd-initial** start.vert

V-AUS restart file (.dat)

This file is required for continuing a finished V-AUS job. It should be specified in the system configuration file:

- –fn-i-aus-restart aus_rest.dat

At the end of a V-AUS job, this file is automatically generated at the file, specified by *–fn-o-aus-restart* option.

Atom group definition file (.inp)

This file defining groups of atoms. This informations are used for:

- Canceling the center of mass motion
- Defining enhaced groups for V-AUS simulations

In this ascii file, each line defines one atom group. The characters at the head of each line indicate the name of each group. Successive columns specify atoms in this group.

For example:

```
group1 1 4 6-9
group2 3 10-11 13
```

The group1 is composed of atoms 1, 4, 6, 7, 8, and 9. The group2 is composed of atoms 3, 10, 11, and 13.

The atom-ID are began from 1.

Output files

1. Standard output
2. Trajectory file (.cod)
3. Restart file (.restart)
4. V-McMD (or V-AUS) lambda trajectory
5. V-McMD (or V-AUS) virtual-system trajectory
6. V-AUS restart file

A simulation log will be output for the standard output. Redirection to a file is recommended.

The trajectory file format is compatible to myPresto/Psygene.

This file repeats the two pars: a header of the frame, and atomic coordinates at the frame.

For the header part:

- [4 bytes, INT] The size of header parts in bytes. Always “44”.
- [4 bytes, INT] Step number
- [4 bytes, FLOAT] Time
- [4 bytes, FLOAT] CPU time
- [4 bytes, FLOAT] Total energy
- [4 bytes, FLOAT] Kinetic energy
- [4 bytes, FLOAT] Temperature
- [4 bytes, FLOAT] Potential energy
- [4 bytes, FLOAT] Always “0.0”
- [4 bytes, FLOAT] Always “0.0”
- [4 bytes, FLOAT] Always “0.0”
- [4 bytes, FLOAT] Always “0.0”
- [4 bytes, INT] The size of header parts in bytes. Always “44”.

For the coordinates part:

- [4 bytes, INT] The size of this part. The nubmer of atom * 3 dimensions * 4 bytes.
- [FLOAT] X, Y, and Z coordinates of each atoms.
- [4 bytes, INT] The size of this part. The nubmer of atom * 3 dimensions * 4 bytes.

The restart file format is compatible to myPresto/Psygene.

This file is composed of the three pars: a header of the frame, atomic coordinates, and velocities.

For the header part:

- [4 bytes, INT] The length of the following text. Alwasy “80”.
- [80 bytes, CHAR] Description of this simulation (version information)
- [4 bytes, INT] Alwasy “80”.
- [4 bytes, INT] Alwasy “8”.
- [4 bytes, INT] The number of atoms for coordinates.
- [4 bytes, INT] The number of atoms for velocities.
- [4 bytes, INT] Alwasy “8”.
- [4 bytes, INT] Alwasy “36”.

- [4 bytes, INT] Step number.
- [4 bytes, FLOAT] Time.
- [4 bytes, FLOAT] Total energy.
- [4 bytes, FLOAT] Kinetic energy.
- [4 bytes, FLOAT] Potential energy.
- [4 bytes, INT] Alwasy "36".

For the coordinates part:

- [4 bytes, INT] The size of this part in bytes. The number of atoms * 3 dimenstions * 8 bytes.
- [DOUBLE] X, Y, Z coordinates of each atom.
- [4 bytes, INT] The size of this part in bytes. The number of atoms * 3 dimenstions * 8 bytes.

For the velocity part:

- [4 bytes, INT] The size of this part in bytes. The number of atoms * 3 dimenstions * 8 bytes.
- [DOUBLE] X, Y, Z velocities of each atom.
- [4 bytes, INT] The size of this part in bytes. The number of atoms * 3 dimenstions * 8 bytes.

For V-McMD or V-AUS simulations, the lambda values are written on this file.

When *–format-o-extended-lambda ascii* is specified, a lambda value is recorded in each line of the ascii file.

When *–format-o-extended-lambda binary*, is specified, the values are dumped as a binary file.

- [1-4 bytes] The magic number
- [5-8 bytes] The precision (4 or 8)
- [9-14 bytes] Always 1.
- [After that] The lambda values

The trajectory of virtual-system coordinates is written as a two-columns, tab separated table.

- The first column means the step number.
- The second column means the virtual-system coordinate.

For example, in the case that virtual-system transitions are done in every 1000 steps,:

```
1     1
1001  2
3001  1
4001  2
5001  3
```

1.1.3 Analysis

Author Kota Kasahara

Log

Calculation log is printed to the standard output. The total energy of the system should be constant, in the micro-canonical ensemble. When the total energy drifts, you should take care of the following points:

1. Increase –nsgrid-cutoff
2. Decrease –nsgrid-update-intvl
3. Increase –cutoff

```
Step:        0    Time:     0.0000
Total:     -7.4102976562e+04
Potential: -8.7917343750e+04    Kinetic:  1.3814368164e+04
Bond:       2.2974748345e+02    Angle:    9.5691961023e+01
Torsion:    2.9869403994e+02    Improper: 2.9828235618e+00
14-VDW:     1.0501907707e+02    14-Ele:   1.8181411150e+03
VDW:        1.5527498751e+04    Ele:      -1.0599511706e+05

Step:   100000    Time: 50000.0000
Total:     -7.4026648438e+04
Potential: -8.7759882812e+04    Kinetic:  1.3733237305e+04
Bond:       9.4730405123e+03    Angle:    2.4954034353e+02
Torsion:    3.2906814547e+02    Improper: 1.2255384929e+01
14-VDW:     1.2528642296e+02    14-Ele:   1.8064233115e+03
VDW:        1.7495527442e+04    Ele:      -1.1725102074e+05
```

Trajectory

omegagene output the trajectory in the format compatible with myPresto/psygene-G. If you want to convert the trajectory file into Gromacs .trr format, the script to do it is included in *omega-toolkit*.

python ${OMEGATK}/convert_trajectory_presto.py –i-pdb initia.pdb –i-crd traj.crd -o traj.trr

1.1.4 Samples

Author Kota Kasahara

Sample files

Some sample simulation data are attached.

In following instruction, it is assumed that the variable $OMEGATK indicates the path to the *omega_toolkit*. And the binary *omegagene* is placed in a directory in ${PATH}, and the library files *libCelesteCUDA.so* and *libCelesteCore.a* are placed in ${LD_LIBRARY_PATH} directories.

NVE calculation (Trp-cage)

Move to the directory *cal01_nve/run00001*.

And execute the bash file:

```
bash 1_generate_velocities.bash
```

In this shell, the initial velocities are generated. See the output:

```
number of atoms: 13277
Number of rigid units: 4495
TEMPERATURE:300.0
[ 2.05517825e-11 4.44033144e-11 -9.32866839e-11]`
```

The last line indicates the momentum of COM in each axis.

Next, check the simulation setting in *system.cfg* and *atom_groups.inp*:

```
vi system.cfg
vi ../atom_groups.inp
```

Generate the input binary file:

```
bash 2_mdinput_gen.bash
```

The log is output in *mdinpgen.out*:

```
size: buf_pos_rest: 4
size: buf_group_coord: 0
End
```

Then, execute omegagene. Before that, check the setting file *md.cfg*:

```
vi md.cfg
```

Execute it:

```
bash 3_submit.bash
```

The directory *run00002* is for test of restarting the calculation in *run00001*:

```
cd ../run00002
diff system.cfg ../run00001/
diff md.cfg ../run00001/
bash 4_mdinput_gen_2.bash
bash 5_submit_2.bash
```

Check the accordance of the energy values at the last of *run00001* and the first of *run00002*.

Other ensembles

Some other samples can be performed in a similar way.

- *cal02_nvt* for NVT calculation (Trp-cage)
- *cal03_vmcmd* for V-McMD calculation (Trp-cage)
- *cal04_aus* for V-AUS calculation (2 Alanine-pentapeptides)

1.2 Celeste Build/Installation Documentation

Author Benson Ma

Contents:

1.2.1 Installation

Author Kota Kasahara

Contents

Celeste is written in C++, and its build system utilizes CMake. The following is a non-exhaustive description for building Celeste. Currently there are three compile options of Celeste:

1. CPU
2. CPU without the neighbor search algorithm (inefficient)
3. CPU with GPU (CUDA)

For platform-specific details on building Celeste, please refer to the *Celeste Build Notes*.

Software Requirements

- CMake 3.4+
- For the GPU version of Celeste, CUDA 7.0+ is required for C++11 support.
- **A C++11 compiler:**
  - GCC: 4.8+
  - Clang: 3.6+
  - AppleClang: 5.0+
  - Intel: 15.0+ (minimal version required by CUDA 7.0+)
- OpenMP 3.1+
- Python 2.7.x
- numpy

Building Celeste - Standard Version

1. Set up a target build folder:

```
# in <PROJECT_ROOT> directory
localhost:celeste local$ mkdir target
localhost:celeste local$ cd target
```

2. Configure the build. CMake will determine all the external software dependencies for the selected build variant, and exit with errors if the dependency requirements are not met. CMake must be invoked on the `CMakeLists.txt` file in the <PROJECT_ROOT> directory:

```
# in <PROJECT_ROOT>/target directory
localhost:target local$ cmake ..
```

3. Build the software:

```
# The verbose flag is optional
localhost:target local$ make VERBOSE=1
```

Building Celeste Without Neighbor Search Routines

While neighbor search is effective for fast calculations, the implementation is complicated and may be difficult to debug MD runs. For this reason, a version of Celeste without the neighbor search routines can be built for debugging or testing.

To build this version of Celeste, simply run the following command instead when configuring the build (Step 2):

```
localhost:target local$ cmake -DCELESTE_WO_NS=1 ..
```

The compiled executable will be named `celeste_wons`.

Building Celeste With GPU Acceleration

For building this version of Celeste, CUDA 7.0+ is required. For running the binary, an NVIDIA GPU with Compute Capability >= 3.5 or later is required.

To build this version of Celeste, simply run the following command instead when configuring the build (Step 2):

```
localhost:target local$ cmake -DCELESTE_GPU=1 ..
```

CMake will automatically determine the default installation paths for the CUDA libraries and `nvcc`. Please refer to the Build Notes if you have installed CUDA to a custom filesystem path.

The compiled executable will be named `celeste_gpu`.

Celeste Toolkit

CelesteTookit is a library of pre- and post-processing scripts for MD simulations to be used with Celeste. It requires Python 2.7.x and the `numpy` library.

This manual assumes that the CelesteToolkit directory specified in the environmental variable `${CELESTETK}`. This path should be added in `${PYTHONPATH}`:

```
export CELESTETK="${HOME}/celeste/toolkit"
export PYTHONPATH=${CELESTETK}:${PYTHONPATH}
```

```
setenv CELESTETK "${HOME}/celeste/toolkit"
setenv PYTHONPATH ${CELESTETK}:${PYTHONPATH}
```

1.2.2 Celeste Build Notes

Author Benson Ma

Contents

Below is an assortment of build notes for handling different software dependencies and different platforms.

General

CUDA

- Library code that is built on top of CUDA must be built as **shared libraries**; otherwise, linker errors will appear during building on Linux platforms. Hence, the entries in `CMakeLists.txt` specifying building CUDA-dependent libraries should be marked `SHARED` as such:

```
CUDA_ADD_LIBRARY(CelesteFooCUDA SHARED foo.cu bar.cu)
```

- The version of gcc installed may be a later version than the the latest officially-supported host compiler for `nvcc`. You will see an error like this:

```
# in <PROJECT_ROOT>/target
localhost:target local$ cmake -D CMAKE_C_COMPILER=/opt/local/bin/gcc-mp-5 -D␣
 ↪CMAKE_CXX_COMPILER=/opt/local/bin/g++-mp-5 -D CELESTE_GPU=1 ..
...
... generating Makefiles
...

localhost:target local$ make
...
... building code
...
/usr/local/cuda/include/host_config.h:115:2: error: #error -- unsupported GNU␣
 ↪version! gcc versions later than 4.9 are not supported!
```

  While not recommended, this can be fixed by commenting out the appropriate `#error` macro in `<CUDA_ROOT>/include/host_config.h`:

```
#if __GNUC__ > 4 || (__GNUC__ == 4 && __GNUC_MINOR__ > 9)

// #error -- unsupported GNU version! gcc 4.10 and up are not supported!

#endif /* __GNUC__> 4 || (__GNUC__ == 4 && __GNUC_MINOR__ > 9) */
```

Linux

MPI

- Installation of MPICH or OpenMPI may not include adding `mpicc`/`mpic++` to the `$PATH`, resulting in the following error when `cmake` is invoked:

```
Could NOT find MPI_C (missing: MPI_C_LIBRARIES MPI_C_INCLUDE_PATH)
```

  To resolve this, simply add the directory containing `mpicc`/`mpic++` to the `$PATH` in the `ENVIRONMENT` or in the *cmake*‘ invocation:

```
localhost:target local$ PATH=$PATH:/usr/lib64/mpich/bin cmake ..
```

Mac OS X

- Unfortunately, with the newer versions of CUDA on the Mac, gcc is not a supported host compiler. Attempting to compile CUDA code using gcc as the host compiler will result in an error message that looks like this:

```
# in <PROJECT_ROOT>/target
localhost:target local$ cmake -D CMAKE_C_COMPILER=/opt/local/bin/gcc-mp-5 -D␣
 ↪CMAKE_CXX_COMPILER=/opt/local/bin/g++-mp-5 -D CELESTE_GPU=1 ..
...
... generating Makefiles
...

localhost:target local$ make
...
... building code
```

```
...
nvcc fatal   : GNU C/C++ compiler is no longer supported as a host compiler on␣
 ↪Mac OS X.
```

Intel's ICC does not appear to be a supported host compiler for CUDA on Mac OS X either. Only Clang appears to be a supported host compiler, but this only applies to "AppleClang" (the version of Clang maintained by Apple). Attempting to use (newer versions of) mainline Clang will result in an error message that looks like this:

```
# in <PROJECT_ROOT>/target
localhost:target local$ cmake -D CMAKE_C_COMPILER=/opt/local/bin/clang-mp-3.7 -D␣
 ↪CMAKE_CXX_COMPILER=/opt/local/bin/clang++-mp-3.7 -D CELESTE_GPU=1 ..
...
... generating Makefiles
...

localhost:target local$ make
...
... building code
...
nvcc fatal   : The version ('30700') of the host compiler ('clang') is not␣
 ↪supported
```

While not recommended for ABI/linking reasons, issues such as this above can be resolved by specifying a _different_ compiler as the host compiler for `nvcc`:

```
# where /usr/bin/clang symlinks to AppleClang
localhost:target local$ cmake -D CMAKE_C_COMPILER=/opt/local/bin/clang-mp-3.7 -D␣
 ↪CMAKE_CXX_COMPILER=/opt/local/bin/clang++-mp-3.7 -D CELESTE_GPU=1 -D CUDA_
 ↪HOST_COMPILER=/usr/bin/clang ..
```

Windows

- MSVC does not define the alternative tokens for logical operators (i.e. and in place of &&) by default. See http://stackoverflow.com/questions/24414124/why-does-vs-not-define-the-alternative-tokens-for-logical-operators. This issue can be circumvented by including the following header in source files that use alternative tokens:

  ```
  #include <ciso646>
  ```

  The correct solution is to disable C++ language extensions in MSVC by use of the `/Za` compiler flag; however this flag is known to be buggy and will result in ODR errors during linking. See the following articles:

  - http://cidebycide.blogspot.com/2015/10/visual-studio-2015-icu-and-error-lnk2005.html
  - http://stackoverflow.com/questions/31808256/multi-file-iostream-error-lnk2005-in-vs2015-with-za

First-time users are advised to consult the *Omegagene Users' Documentation* first to learn how to get started with omegagene.

CHAPTER

TWO

ABOUT OMEGAGENE

"omegagene" is the name of the molecular dynamics (MD) simulation software. omegagene has several unique features, such as, the zero-multipole summation method and the virtual-system coupled multi-canonical MD method.

The current version of omegagene can perform:

1. MD simulations on NVE ensemble
2. MD simulations on NVT ensemble (The velocity rescaling, or Hoover-Evans thermostat)
3. MD simulations on Multi-canonical ensemble, with the Virtual-system coupled McMD method.
4. Applying constraints with SHAKE algorithm
5. Calculations of electrostatic potentials based on the zero-dipole summation method.
6. Calculations of pairwise potentials on GPGPU (powered by CUDA 7.0, Computer Capability 3.5 is required).

CHAPTER

THREE

ABOUT CELESTE

CELESTE is the code name for the collection of libraries that build omegagene

CHAPTER

FOUR

DEVELOPERS

- KASAHARA Kota, Ritsumeikan Univ., Japan
- MA Benson, Univ. Illinois, US
- GOTO Kota, TITECH, Japan
- BHASKAR Dasgupta, Osaka Univ., Japan
- HIGO Junichi, Osaka Univ., Japan
- FUKUDA Ikuo, Osaka Univ., Japan
- MASHIMO Tadaaki, AIST, Japan
- AKIYAMA Yutaka, TITECH, Japan
- NAKAMURA Haruki, Osaka Univ., Japan

4.1 Indices and tables

- genindex
- modindex
- search