MC^{3}S

Monte Carlo for Complex Chemical Systems

User Manual

Version 2.2

January 2010

Comments or questions: beut0013@umn.edu

1 Introduction

This manual is intended to explain the variables necessary for simulations carried out in the canonical (NVT), isobaric-isothermal (NpT), NVT-Gibbs or NpT-Gibbs ensemble. Simulations in the grand canonical (μVT) ensemble are possible but not discussed. The Lennard-Jones 12-6 potential is assumed for non-bonded interactions. Other functional forms, such as exponential-6, 9-6 or mmff, have been implemented but should be considered non-generalized and used with caution.

2 Revising the code

For clarity and to assist future users, please bracket all changes to the code with the following lines:

- c — [*your initials*] (*date*)
- c — [*explain what changes were made or the purpose of the new code*]
- *new or revised code here*
- c — END [*your initials*] (*date*)

3 Compiling and running the code

The many files that comprise the MCCCS code are located in a directory titled **MCCCS**. The newest version can be downloaded from the internal group website. The simulation can be started after compiling the code and setting up the necessary input files.

Before compiling the code, check to ensure that all logical parameters in **control.inc** conform to the desired system. Changes in any **.inc** file will not take effect until the code is (re)compiled. To compile the code, choose the appropriate compiler in **Makefile** and execute the command **make** while in the directory that contains the MCCCS code. If desired, change the name of the executable in **Makefile** before compiling. To completely rebuild the code from the source, execute **make clean** followed by **make**.

Sample input files are located in a sub-directory within the **MCCCS** directory. The input file(s) (**fort.4** and possibly others) should be placed in a different directory from the code. After setting all necessary variables in **fort.4** and verifying that any other necessary input files are included, navigate to the MCCCS directory and type the name of the executable (given in **Makefile**) to run the code.

4 Input File: fort.4

This is the main input file for a simulation. It contains information about the system to be simulated, including the number and type of molecules, how these molecules are to be modeled and what type of moves will be utilized. An error from the **readdat.f** file often indicates a problem in the **fort.4** input file. Note that most variables, even if they are not used in a particular simulation, must be present in **fort.4** at all times; however, there are a few variables that must be removed depending on the setting of other variables. This section specifies those variables that are sometimes removed; if not specified, assume that a particular variable must be included in **fort.4**.

4.1 System Specification

seed: Initializes the random number generator with an integer seed value. If the seed stays the same, the program will always generate the same sequence of pseudo-random numbers, which is useful for debugging. Change the seed to any other integer to generate a different sequence.

iounit: Set to 2 to print to a file. Set to 6 to print to the screen.

L_add: Used to add particles to the system to increase a box to the desired size. The particles may be ghost atoms, helium atoms, L-J particles or any other particles with a single interaction site. May be used instead of manually adding particles to the fort.77 file. If true, **L_add** adds particles to the system. Usually false.

N_add: The number of particles to be added to the system.

N_box2add: The box to which the particles will be added.

N_moltyp2add: The molecule type of the added particles. The particles must have a single interaction site.

L_sub: Used to remove particles from the system to decrease a box to the desired size. The particles may be ghost atoms, helium atoms, L-J particles or any other particles with a single interaction site. May be used instead of manually removing particles from the fort.77 file. If true, **L_sub** removes particles from the system. Usually false.

N_sub: The number of particles to be subtracted from the system.

N_box2sub: The box from which the particles will be subtracted.

N_moltyp2sub: The molecule type of the subtracted particles. The particles must have a single interaction site.

lecho: Echos the data read in by **readdat.f** to the standard output. Should be set to true.

lverbose: Yields verbose output of the input variables if true.

l_Coul_CBMC: If true, uses the Coulombic energy during the Boltzmann weight calculation. Normally true.

Num_cell_a: The number of unit cells in the **a** direction for solid simulations.

Num_cell_b: The number of unit cells in the **b** direction for solid simulations.

Num_cell_c: The number of unit cells in the **c** direction for solid simulations.

run_num, suff: The run number and suffix are included as identifiers in the names of the output files. In this manual, references to files such as **config##.dat** indicate a file whose name includes the **run_num** and **suff**.

nstep: The number of Monte Carlo cycles (steps) to run the simulation.

lstep: If true, **nstep** refers to MC steps. If false, **nstep** refers to cycles. This statement controls all references to "cycles" or "steps" and should be false unless debugging.

lpresim: If true, runs a presimulation to generate the SAFE-CBMC probabilities. There must be only one particle in the simulation and **pmfix** must be set to 0. If SAFE-CBMC probabilities are required for multiple particle types, run a presimulation for each particle type and combine the probabilities in the **fort.23** file before beginning the simulation with multiple types of particles.

iupdatefix: Specifies how often the SAFE-CBMC probabilities should be updated.

L_tor_table: For the torsional potential. If true, uses tabulated potentials instead of the regular functional form.

L_spline: Set to true for spline interpolation of the torsional potential. It has no meaning if **L_tor_table** is false.

L_linear: Set to true for linear interpolation of the torsional potential. It has no meaning if **L_tor_table** is false.

L_vib_table: Turns on tabulated vibrations. Requires **fort.41**. All equilibrium bond lengths must be listed in **suvibe.f** and all force constants must be set to zero.

L_bend_table: Turns on tabulated 1-3 bending. Requires **fort.42**. 1-3 interactions must be specifically included (see **internal 1-4**).

L_vdW_table: Turns on the tabulated van der Waals potential. Requires **fort.43**. All bead types must be included in **suijtab.f**.

L_elect_table: Turns on tabulated electrostrostatics. Requires **fort.44**. All bead types must be included in **suijtab.f**. Currently, the tabulated potential is multiplied by the partial atomic charges within the code, which works nicely with Greg Voth's force-matched potential, but this may change.

nbox: The number of simulation boxes. Note that some variables, including **boxlx, boxly, boxlz, lsolid, lrect, kalp, rcutchg** and **rcutnn**, must be specified for each box.

express: The external pressure, in MPa, for each box in an NpT or NpT-Gibbs ensemble simulation.

ghost_particles: The number of ideal gas particles offset for each box. Usually set to zero. There must be **nbox** values. The offset particles do not interact with the system via the interaction potential; rather, they contribute only to the ideal gas part of, for example, pressure. Offset particles can be used when an ideal or near-ideal gas is needed as a reference

state to compute, for example, the free energy of transfer of a particle molecule between an ideal gas and water.

L_Ewald_Auto: For the automatic assignment of Ewald parameters. If true, the code will generate **kalp(ibox)** and **rcutchg(ibox)** based on the system size. **kalp** is calculated by dividing 3.2 by **rcut**. This is the standard method but always check for convergence of the Ewald summation. Remember to set **lewald** in **control.inc** to true to turn on the Ewald summation.

temp: The temperature in K.

fqtemp: The fluctuating charge temperature, in K, for simulations using the ANES-MC or polarizable models.

Elect_field: The electric field strength, in units of V/Å, for each simulation box. The electric field is applied in the z-direction. Note that **lelect_field** must be set to true in **external.inc**.

ianalyze: The analysis subroutine is called every **inanalyze** cycles. Could be a number smaller than **imv** to yield better statistics.

nbin: The number of bins for the radial distribution function calculation. A value of 200 should be adequate for most purposes.

lrdf: If true, the radial distribution function (rdf) will be calculated.

lintra: If true, intramolecular distances will be included in the rdf.

lstretch: If true, the bond stretch distribution will be calculated.

lgvst: If true, *gauche* versus *trans* statistics will be calculated.

lbend: If true, the bending distribution will be analyzed.

lete: If true, the end-to-end distribution of flexible chains will be calculated. Flexible chains should be numbered so that the first bead is at one end of the chain and the last bead is at the other end.

lrhoz: If true, Z profiles (for solid slabs or self-assembled monolayers) will be calculated.

bin_width: Specifies the bin width for the end-to-end distribution and density profile. A value of 0.2 is adequate for most purposes.

iprint: The simulation status will be printed every **iprint** cycles (steps). Can be used to track the progress of a simulation as it is running. A useful value yields 10-20 outputs per simulation.

imv: The system configuration is sent to the movie file, **movie##.dat**, every **imv** cycles (steps). As this is used for subsequent analysis, it is important to have reasonably statistically independent results. Larger values of **imv** (for example, greater than 500) will yield results that are less correlated (more statistically independent).

iratio: The maximum translation and rotation values are updated every **iratio** cycles (steps). Smaller values (approximately 250-500) are acceptable for equilibration. For production, **iratio** must be greater than **nstep** to avoid violating microscopic reversibility.

iblock: Indicates the number of cycles (steps) to be independently averaged when calculating properties such as p, ρ or molefraction. At least 1000 cycles per block is a good value. A constant drift in the block average of a given property (for instance, steadily decreasing energy values) indicates that the system is not yet equilibrated.

idiele: For dielectric constant calculations, the system dipole is calculated every **idiele** cycles (steps) and the x, y and z components are written to **fort.27**. The dielectric constant needs to be calculated separately using **fort.27** (see description of **fort.27** in section 5.2). For single-component NpT simulations, **idiele** also controls the frequency of the output volume and energy statistics as output in **fort.14** through **fort.19**.

L_movie_xyz: Yields a single movie file in xyz format if true.

iheatcapacity: If **iheatcapacity** is less than **nstep** and **lnpt**, which is in **control.inc**, is false, then **fort.55** contains the following averages: $< E >$ and $< E^2 >$. If **iheatcapacity** is less than **nstep** and **lnpt**, which is in **control.inc**, is true, then **fort.56** contains the following averages: $< H >$ and $< H^2 >$.

Note that the next eight variables (**boxlx** through **rcutnn**) must be included for each box in the system. All eight variables for the first box should be listed first, followed by sets of these eight variables for subsequent boxes, if applicable.

boxlx, boxly, boxlz: Specifies the x, y and z dimensions, in Å, of the simulation box. Used only for initialization. Note that the box length must be greater than twice the cutoff (**rcut**). If, for example, **rcut** is 14 Å, it is a good idea to start with a box length of slightly more than 2**rcut** (for example, 32 Å) to ensure no volume moves that would reduce the box length to less than 28 Åwill be attempted. If such a move is attempted, the simulation will be aborted.

lsolid: If true, the x, y and z box lengths can vary independently. Only to be used if the box can withstand anisotropic stressors (*i.e.*, if the box is crystalline).

lrect: If **lsolid** and **lrect** are both true, then the cell will remain rectangular (*i.e.,* the axes will remain mutually perpendicular). This variable has no effect if **lsolid** is false. Note that in the case of a non-cubic, non-rectangular system, **linit** must be false and the system must start from a **fort.77** file.

kalp: This is the Ewald sum convergence parameter. It will be calculated automatically as 3.2 divided by **rcut** if **L_Ewald_Auto** (described above) is true. If **L_Ewald_Auto** is false, the value of the Ewald sum convergence parameters will be taken from this variable. If all charge interactions are being calculated (*i.e.,* if **lchgall** in **control.inc** is true), then **kalp** should be greater than or equal to 5.6. If a charge cutoff is being used (*i.e.,* if **lchgall** is false), then **kalp** should be greater than or equal to 3.2 divided by **rcut**. Verification of the convergence of the Ewald sum is highly recommended.

rcut: This is the cutoff of the Lennard-Jones potential. Interactions beyond **rcut** are not calculated. A smaller value might be used for equilibration than production. The TraPPE force field uses a 14 Å-cutoff. In the simulation of a large vapor box with the Ewald sum, it is efficient to set **rcut** to be 40% of **boxlx**. This reduces the size of **kalp**, which in turn reduces the number of vectors required (controlled by **vectormax** in **control.inc**).

rcutnn: The nearest neighbor bead-bead cutoff for the neighbor list. Not normally used.

nchain: The total number of molecules in the simulation.

nmolty: The number of different molecule types in the simulation.

moltyp: The number of molecules of each type. Note that there must be **nmolty** values.

lmixlb: If true, the Lorentz-Berthelot combining rules for the Lennard-Jones potential will be used. These are the rules used by the TraPPE force field and what should be used by default.

lmixjo: If true, the Jorgensen combining rules for the for the Lennard-Jones potential will be used. These are the rules used by the OPLS force field.

nijspecial: This variable specifies whether to adjust the combining rules by a scaling parameter. **nijspecial** lists the number of special L-J interactions. If **nijspecial** is 0 then no scale parameter is used and **ispec, jspec, aspec** and **bspec** are ignored. If **nijspecial** is 1 or greater, **aspec** and **bspec** modify the combining rules using $\sigma_{ij} = \textbf{aspec} \cdot \sigma_{ij}$ and $\epsilon_{ij} = \textbf{bspec} \cdot \epsilon_{ij}$.

ispec, jspec: Specifies which parameters (as listed in **suijtab.f**) should be modified by the scale factors **aspec** and **bspec**. Used only if **nijspecial** is greater than or equal to 1.

aspec, bspec: The scale factors used to modify the parameters denoted by **ispec** and **jspec**. **aspec** modifies the combined σ_{ij} value. **bspec** modifies the combined ϵ_{ij} value. Used only if **nijspecial** is greater than or equal to 1.

rmin: The minimum cutoff for atom-atom distances. Any move that would bring atoms closer than this distance will be automatically rejected. In theory **rmin** can be set to any value as long as $g(r)$ is zero up to that distance. A smaller value, even 0 Å, may be used for equilibration than production. A good value for a system that contains hydrogen bonds is 1.2 Å. A good value for a system without hydrogen bonding is 0.7σ.

softcut: The upper bound on numbers of which to take the negative exponential. Used to rule out events with low probability. 100.0 is a reasonable value.

rcutin: Specifies the inner cutoff for dual-cutoff CBMC. During CBMC growths, only intermolecular interactions between atoms within **rcutin** of one another are calculated. Smaller values of **rcutin** will require less computer time while larger values will have higher acceptance rates. Typical values are between 5 and 9 Å.

rbsmax, rbsmin: Specifies the maximum and minimum radii of the "in" region during an AVBMC move. These values are dependent on the type of molecule. Typical values for an alcohol may be 5 Å and 3 Å, respectively.

4.2 Molecule Specification

Each type of molecule must have its own specification section in which the connectivity and potential parameters are specified. Note that a "bead" is one unit of a molecule. It may contain a single atom or a group of atoms. For example, in a united atom model, a CH_x group is considered to be a single bead.

Certain types of molecules should be numbered in specific ways to avoid errors or to increase efficiency. In a partially rigid molecule, the flexible part must be numbered first. For example, only the nitro group in nitrotoluene is flexible. The two oxygen beads are given numbers 1 and 2 and the nitrogen atom is assigned bead number 3. There is one growpoint at bead number 3.

For branched molecules, it is often best to start numbering at the most highly branched site. For example, for methyl acrylate, the carbonyl carbon should be bead number 1 and the beads to which it is bonded should be numbers 2, 3 and 4.

4.2.1 Molecule Parameters

nunit: The number of beads in the molecule.

nugrow: The maximum number of beads to regrow with CBMC. Should be equal to **nunit** unless **explct.f** is used.

ncarbon: For the explicit hydrogen model, the number of carbon atoms present. This allows for a simplified setup of the exclusion table. Note that these atoms should be listed first in the molecule specification section of the **fort.4** file. **ncarbon** should be equal to **nunit** for all other models.

maxcbmc: The maximum number of beads that are allowed to grow with CBMC. Should be equal to **nugrow**.

iurot: For a single rotation center, **iurot** specifies the bead around which to perform rotations. 0 specifies the center of mass and should be used as the default. For multiple rotation centers, set **iurot** < 0 and add values for **nrotbd**, **irotbd** and **pmrotbd** immediately after all of the variables in section 4.2.3 (immediately after **pmbias2**). If **iurot** ≥ 0, then these three variables are not included.

lelect: If true, the molecule contains one or more charged sites.

lflucq: If true, the charge(s) on molecules of this type may fluctuate. Only for polarizable models.

lqtrans: If true, intermolecular charge transfers for molecules of this type are allowed. For polarizable molecules.

lexpand: If true, this molecule will be treated with the expanded ensemble. The default setting is false.

lavbmc1, lavbmc2, lavbmc3: Logical variables to specify which of the three types of AVBMC moves to perform on this molecule. Used only if the intrabox swap move is specified (see **pmswap**).

fqegp: Specifies an energy offset in the fluctuating charge coulombic energy. Only for polarizable models.

maxgrow: The maximum number of interior segments to consider in a SAFE-CBMC regrowth (see **pmfix**).

lring: Specifies whether the molecule is a flexible ring that should be regrown with SAFE-CBMC during a swap move. If true, **iring** must be specified on the following line.

lrigid: If true, part or all of this molecule should be treated as a rigid body. Note that if **lrigid** is true, **growpoints** must be included. If **lrigid** is false, **growpoints** must not be included. Typically **lrigid** and **lbranch** (see below) are both true or both false.

lrig: If true, rigid segments should be grown from non-rigid segments during SAFE-CBMC. The default setting is false. If true, additional variables must be specified after this line.

nrig: Only included if **lrig** is true. If **nrig** is less than or equal to 0, the site from which to grow the rigid sites will be chosen randomly. If **nrig** is greater than 0, **nrig** specifies the number of specific points to be kept rigid during a SAFE-CBMC growth.

irig: Only included if **lrig** is true and if **nrig** is greater than 0. Specifies the site from which the rigid part will be grown. Must be **nrig** pairs of **irig** and **frig**.

frig: Only included if **lrig** is true and if **nrig** is greater than 0. Specifies the site prior to **irig**. The site denoted by **frig** is not kept rigid. Must be **nrig** pairs of **irig** and **frig**.

nrigmin: Only included if **lrig** is true and if **nrig** is less than or equal to 0. This variable denotes the minimum amount of the chain to keep rigid.

nrigmax: Only included if **lrig** is true and if **nrig** is less than or equal to 0. This variable denotes the maximum amount of the chain to keep rigid.

lsetup: If true, try to automatically setup the bead-bead potential types. Only for linear molecules. Experimental. Use with caution.

isolute: Specifies how often to write the configuration of this molecule to the **fort.11** for later analysis. Identical to **imov** but specific to this type of molecule. Unless specifically needed, a value larger than **ncycles** is recommended.

eta: Specifies any additional potential energy to be added uniformly to each molecule of this type in a particular simulation box. Usually 0. There must be one value for each simulation box.

lq14scale: If true, 1-4 charge-charge interactions need to calculated. Usually true for the TraPPE force field. True for acrylates and diols.

qscale: If **lq14scale** is true, **qscale** gives the scaling factor for the 1-4 charge-charge interactions. Generally 0.5.

growpoints: Specifies the number of sites in a partially rigid molecule from which CBMC growths should be attempted. Only needed if **lrigid** is set to true. Note that an error will occur if **growpoints** is included while **lrigid** is false. If non-zero, **growpoints** must be followed by the unit numbers of the growth sites with one site listed per line.

4.2.2 Bead Specification and Connectivity

The variables in this section must be included for each bead in the molecule.

unit: The bead number.

ntype: This value specifies the interaction parameters of this particular bead as given in the file **suijtab.f** (see section 8 below).

leaderq: This value specifies a neutral charge group of which **unit** is a part. It is used only if **lchgall** is false. It should be set to the lowest unit number in the neutral charge group or otherwise to **unit**.

vibration: This section specifies the bonds formed by this bead. The first line is the number of bonds and is followed by one line for each bond that contains two values: first the unit number to which **unit** is bonded and then the bond type as given by the **suvibe.f** file.

bending: This section specifies the bond-bending interactions that include **nunit**. The organization of this section is similar to the structure of **vibration**: the first line specifies the number of bond-bending interactions and the following lines specify the two other units involved in the bend (the unit closest to **unit** should be listed first) followed by the bend type as given by the **suvibe.f** file.

torsion: This section specifies the torsional interactions that include **nunit**. The organization of this section is similar to the structure of **vibration** and **bending**: the first line specifies the number of torsional interactions and the following lines specify the three other units involved in the torsion in order of decreasing proximity to **unit** followed by the torsion type given by the **suvibe.f** file.

4.2.3 AVBMC Variables

If AVBMC moves are to be performed (that is, if **lavbmc1**, **lavbmc2**, or **lavbmc3** is true), the variables listed in this section must be included immediately after **torsion**.

pmbias: This value specifies to probability with which an "out to in" move is attempted. The probability of an "in to out" move is then given by $1 -$ **pmbias**.

pmbsmt: This variable gives the probability with which the target site is a particular molecule type of AVBMC 2 and 3 moves. There must be one value for each molecule type.

pmbias2: This value gives the probability in an AVBMC 3 move that a molecule is taken from the "out" region and moved to a target side. If the molecule is not taken from the "out" region, it is removed from another cluster.

4.2.4 Additional iurot Information

If **iurot** is less than 0, the following three variables must be in the **fort.4** input file. If **iurot** is greater than or equal to 0, these variables are not included.

nrotbd: The number of rotation centers.

irotbd: The unit numbers of the rotation centers. 0 denotes the center of mass. There must be **nrotbd** values.

pmrotbd: The probability to rotate around each center. There must be **nrotbd** values.

4.3 Simulation Box and Move Probability Setup

licell: If true, a link cell list will be used to increase efficiency. This variable should be used only for very large systems and only if **rcut** is less than 0.25 **boxlx**.

rintramax: Specifies the maximum distance between the ends of a molecule for use in the link cell list. One value only (for the largest molecule in the system). Originally for CO_2.

boxlink: Specifies which simulation box to treat with a link cell list.

Armtrax, Armtray, Armtraz: The initial maximum translational displacements, in Å, for all beads in the x, y and z directions. Used to change the conformation and center of mass of a molecule by moving individual bead(s) of that molecule. Only for flexible molecules.

rmtrax, rmtray, rmtraz: The initial maximum translational displacements, in Å, for all molecules in the x, y and z directions. Only used if **linit** is true. If **linit** is false, this information is read from the **fort.77** file.

rmrotx, rmroty, rmrotz: The initial maximum rotational displacements, in radians, for all molecule types around the x, y and z axes. Only used if **linit** is true. If **linit** is false, this information is read from the **fort.77** file.

tatra: The target acceptance ratio for translational moves. 0.5 is a good value. **tatra** is changed every **iratio** cycles (steps).

tarot: The target acceptance ratio for rotational moves. 0.5 is a good value. **tarot** is changed every **iratio** cycles (steps).

linit: If true, the simulation will be initialized. This variable should be true for the initial run and false for all subsequent runs. Note that a **fort.77** input file is required if **linit** is false.

lnewtab: If **lnewtab** is true, a new table of the potential function will be created for zeolite calculations. This variable is normally false.

lreadq: If **lreadq** is true, the bead charge values will be read from the **fort.77** file. If false, the charges will be taken from **suijtab.f**. This is used only for fluctuating charge models.

lbranch: If true, the structure of a molecule must be read from **input_struc.xyz**. If false, the structure can be grown with CBMC and **input_struc.xyz** is not required. Note that one value is required for each molecule type. This variable is usually false unless CBMC cannot grow a particular molecule type. Typically **lrigid** (see above) and **lbranch** are both true or both false.

Note: **inich** through **nchoiq** must be specified for each box in the system. This information is used only for initialization (*i.e.*, if **linit** is true). Even if **linit** is false, however, these variables must be included.

inich: The inital number of molecules of each type in this box. There must be **nmolty** values.

inix, iniy, iniz: Integer numbers of how many molecules should be placed along the x, y and z directions, respectively, in the initial lattice. If the box is cubic, these three values should be relatively close to one another.

inirot: This value specifies the initial rotation along the z axis of each new atom. This applies only to linear molecules and influences the torsional angle. If **inirot** is 0 then each atom is randomly rotated. If **inirot** is greater than 0, each atom is uniformly rotated by **inirot** degrees. If **inirot** is less than 0, the atoms are alternatively rotated in the z direction by $\pm$ **inirot**.

inimix: For mixture simulations, this variable specifies the order in which to select molecules for the initial lattice. If **inimix** is 0, molecules are placed randomly. If **inimix** is greater than 0, molecules are placed in order by type. If **inimix** is less than 0, molecules are placed in alternating order by type.

zshift: This value specifies a uniform shift in the z direction so that terminal beads have a certain z coordinate. Used only for monolayer calculations. **zshift** should be 0 otherwise.

dshift: This value specifies the amount, in Å, to stagger each row in the x direction of the initial lattice. Optimal spacing can be achieved by setting this value to half of the spacing between molecules, which can be calculated dividing **boxlx** by **inix**.

nchoiq: This variable specifies the number of times to call **flucq.f** to optimize the fluctuating charge models in this box and is used only for polarizable models.

rmvol: This value specifies the initial volume displacement. For an NpT simulation a value of 10^3 is appropriate, while 10^{-3} is a good choice for an NVT simulation. The difference is due to different units for the two ensembles.

tavol: Specifies the target acceptance rate for volume moves. Reasonable values for equilibration and production might be 0.4 and 0.5, respectively.

iratv: The maximum volume displacement will be adjusted every **iratv** cycles. **iratv** is analogous to **iratio**, which controls how often the maximum translation and rotation displacements are adjusted. As mentioned for **iratio**, smaller values such as 250-500 are acceptable for equilibration. Note that **iratv** must be greater than **nstep** for production to avoid violating microscopic reversibility.

iratp: The pressure is calculated every **iratp** cycles. As calculating the pressure is somewhat computationally intense, this typically should be set to 5 or greater. A good value is perhaps 10.

rmflcq Specifies the maximum fluctuating charge displacement. For polarizable models only.

taflcq Specifies the target acceptance rate for fluctuating charge moves. For polarizable models only.

4.3.1 Monte Carlo Move Probabilities

The simulation will carry out **ncycles** cycles. Available move types include volume, swatch, swap, CBMC, fluctuating charge, expanded ensemble, translations and rotations; the corresponding probability for each move type is given by **pmvol, pmswat, pmswap, pmcb, pmflcq, pmexpc,** and **pmtra**, respectively. Move probabilities are given cumulatively. For example, if **pmvol** = 0.10 and **pmswat** = 0.25, there is a 10% chance of choosing a volume

move and a $25\% - 10\% = 15\%$ chance of choosing a swatch move. Since the move probabilities are given cumulatively, the probability for a rotation move is simply $1 -$ **pmtra**. Sub-move probabilities, such as on which box to perform a volume move, are also cumulative.

pmvol: Specifies the probability to perform a volume move. Typically start around 0.001 during equilibration and adjust until there is approximately one accepted volume move every 10 cycles.

pmvlmt: The probability to perform a volume move on each box. Must be one probability for each simulation box.

nvolb: For the NVT-Gibbs ensemble, the number of pairs of boxes on which to perform volume exchange moves.

pmvolb: The probability to perform a volume exchange on each pair of boxes in an NVT-Gibbs ensemble. Must be **nvolb** values.

box5, box6: Specifies a pair of boxes for volume exchange. Must be one line for each pair specified in **nvolb**. Not used if the simulation is in the NVT, NpT or NpT-Gibbs ensemble.

pmvolx, pmvoly: These values are required when at least one of the simulation boxes is non-cubic. These values set the (cumulative) probabilities to do moves in the x and y directions, with the z probability being 1 - **pmvoly**. Note that these values must not be included if the system is cubic.

pmswat: Specifies the probability to perform a swatch (CBMC identity exchange) move.

nswaty: The number of pairs of molecule types to perform swatch moves on.

nswatb: Specifies molecule types for the pairs in the swatch move. Must be **nswaty** pairs listed.

pmsatc: Once a swatch move is being performed, this specifies the probability to perform it on a particular pair. Must be **n**swaty values.

nsampos: Specifies the number of beads that can be left in the same positions. **nsampos, ncut, splist, gswatc, nswtcb, pmswtcb,** and **box_pair** must be specified for each pair of molecule types given by **nswaty**.

2xncut: Number of CBMC growth sites the structure given by **n**sampos has to grow back the whole molecule.

splist: Specifies the pair(s) of beads to be kept in the same position. Must be **nsampos:** pairs.

gswatc 2x(ifrom, iprev): For both molecule types, the bead to be grown from and the previous bead must be specified. These will almost always be beads listed in **splist**.

nswtcb: Number of different box pairs to attempt swatch moves between.

pmswtcb: Probabilities to pick a particular box pair for a swatch move. Must be **nswtcb** values.

box_numbers: Specifies the box pairs that this swatch move should be attempted between. Note that this can also list the same box twice, in which case the move is performed inside a single box.

pmswap: Specifies the probability to perform a swap move in which a molecule is removed from one simulation box and placed in another box by growing it with CBMC. It may be efficient to have approximately one accepted swap move per every ten cycles.

pmswmt: This specifies the probability to swap a molecule of a particular type. There must be **nmolty** values.

Note that **nswapb, pmswapb, box1** and **box2** must be repeated **nmolty** times, one for each type of molecule.

nswapb: The number of box pairs on which to perform a swap move on a certain type of molecule.

pmswapb: The probability to perform a swap move on a particular pair of boxes. Must be **nswapb** values.

box1, box2: Specifies the box pairs on which to perform a swap move for this molecule type. If **box1** and **box2** denote the same box, an AVBMC move will be performed, provided **lavbmc1, lavbmc2** or **lavbmc3** has been set to true.

pmcb: Specifies the probability to perform a configurational-bias Monte Carlo (CBMC) move, which involves regrowing a random segment of a molecule in a new configuration in a position near its original position. Typically approximately one-third of the moves in a flexible system are CBMC moves. If dual-cutoff CBMC is desired, remember to set **ldual** in **control.inc** to true (see **control.inc** for more information).

pmcbmt: When a CBMC move is being performed, **pmcbmt** specifies the probability to choose a particular molecule type. There must be **nmolty** values.

pmall: The probability that a particular molecule type should try to regrow itself in its entirety when a CBMC move is performed. Usually set to zero. Must be **nmolty** values.

pmfix: Once a molecule has been chosen for a CBMC move, **pmfix** specifies the probability to perform a SAFE-CBMC move. Usually 0 unless the molecule is large or a flexible ring. Must be **nmolty** values. SAFE-CBMC moves require the **fort.23** input file, which can be generated via a presimulation (see **lpresim** above).

pmflcq: Specifies the probability to perform a fluctuating charge move. For polarizable models only.

pmfqmt: Once a fluctuating charge move has been selected, **pmfqmt** specifies the probability to carry it out on a particular molecule type. Must be **nmolty** values.

pmexpc: Specifies the probability to perform an expanded ensemble move that changes a molecule's interaction parameters.

pmeemt: Once an expanded ensemble move has been selected, **pmeemt** specifies the probability to carry it out on a particular molecule type. Must be **nmolty** values.

pmexpc1: The probability for new expanded ensemble (*i.e.*, to change the identity of the molecule).

pm_atom_tra: The probability to translate individual atoms. Used only in special cases. Usually 0.0d0.

pmtra: Specifies the probability to perform a translation move that changes the position of a molecule. Usually approximately one-third of the total moves are translation moves.

pmtrmt: After a translation move has been selected, **pmtrmt** specifies the probability to carry it out on a particular molecule type. Must be **nmolty** values.

Note that, although not specified, the fraction of rotation moves is equal to 1 − **pmtra**.

pmromt: After a rotation move has been selected, **pmromt** specifies the probability to carry it out on a particular molecule type. There must be **nmolty** values.

nchoi1: Specifies the number of randomly chosen positions to consider during a trial insertion of the first bead during a CBMC swap move. 10 is a reasonable number. Increasing **nchoi1** increases the computer time but also increases the acceptance rate of the swap move. There must be **nmolty** values.

nchoi: Specifies the number of randomly chosen positions to consider during all CBMC regrowths. A larger value takes longer but increases the acceptance rate. 8 is a reasonable value. There must be **nmolty** values. Note that **nchoi** controls both swap and CBMC moves, while **nchoi1** controls only swap moves.

nchoir: Specifies the number of rotations to consider for the swap move of a rigid molecule. Only used if **lrigid** is true for this molecule. There must be **nmolty** values.

nchoih: Specifies the number of explicit hydrogen states to consider during a CBMC regrowth. The value only changes the rotation of the methyl group. Unused for other molecules. There must be one value of **nchoih** for each molecule type.

nchoitor: Specifies the number of torsional angles to consider during a CBMC regrowth. 100 is a good choice. Larger takes longer but increases acceptance rates. There must be one value of **nchoitor** for each molecule type.

nchbna, nchbnb: Specifies the number of bend angles to consider during a CBMC regrowth. 1000 is a good choice. There must be one value of **nchbna** and one value of **nchbnb** for each molecule type. **nchbnb** is only used for branched molecules because, in order to grow two branches at once, it might be necessary to consider more possibilities to find a good bend angle.

icbdir: Specifies whether a CBMC growth should go in only one direction. The usual value, 0, indicates no preference. 1 indicates that CBMC growths are considered only for increasing unit numbers. There must be **nmolty** values.

icbsta: Specifies the starting unit for a CBMC growth. If a consistent starting point is desired for a CBMC move, **icbsta** will specify the unit number. For example, +5 indicates that CBMC growths should always start at bead number 5. A value of -5 indicates that the growth should start at a random position between bead 5 and the highest numbered bead (this is useful when one end of the chain is rigid and tethered to a surface). **icbsta** is usually set to 0, which means there is no preferred starting point. If not set to 0, make sure that the absolute value of **icbsta** + **nugrow** is equal to **nunit** + 1 for consistency. There must be **nmolty** values of **icbsta**.

exclusion: Specifies whether or not to exclude any intermolecular interactions. If no inter-

actions are to be excluded, **exclusion** should be 0 and the following line must be left blank. Otherwise an integer value should be used to indicate the number of pairs of exclusions. For each value, there must be one additional line specifying the molecule type and unit and the molecule type and unit from with it should be excluded from interacting.

internal 1-4: By default, Lennard-Jones interactions are considered only if beads are seperated by four or more bonds and coulombic interactions are considered past three bonds with 1-4 interactions scaled by **q14scale**. **internal 1-4** is an integer variable equal to the desired number of exceptions to this rule. If there are no exceptions, **internal 1-4** should be 0 and the following line must be left blank. If non-zero, there must be that many lines following, each of which gives the molecule type, the two bead numbers to change the rule for, +1 or -1 (for turning the interaction on or off, respectively), the factor to scale the L-J interaction by and the factor by which to scale the coulombic interaction.

oh 1-5 interaction: Specifies whether any additional repulsive interactions should be computed for hydroxyl hydrogen intramolecular oxygen pairs separated by 4 bonds. If there are no additional interactions, **oh 1-5 interaction** should be 0 and the following line must be left blank. Otherwise the first number is the quantity of interactions, followed by one line for each interaction that gives the molecule type, bead numbers and the a15 value, which is 1 is for an ether oxygen or 2 is for an alcohol oxygen.

lucall: If true, the chemical potential is calculated independently. Usually false since the chemical potential is determined automatically by the swap move.

ucheck: If **lucall** is true, **ucheck** indicates for which molecule types the chemical potential should be calculated. A value of 0 means the chemical potential will not be calculated. A value greater than 0 means the chemical potential will be calculated. There must be **nmolty** values.

nvirial: If **lvirial** (in **control.inc**) is true, then the second virial coefficient will be calculated for a molecule. **nvirial** determines the number of configurations to consider at each step. This is a specialized simulation and not for the general case.

start, step: If the second virial coefficient is being calculated, these determine the starting center of mass distance and step size, in Å.

lideal: If true, the box is treated as an ideal gas. The must be **nbox** values. If true, then pair interactions are not computed in the **sumup.f, energy.f, boltz.f** or **atom_energy.f** subroutines. In addition, tail corrections are not calculated (this is accounted for in the **sumup.f, swap.f** and **swatch.f** subroutines). When **lideal** is true, **kalp** for the corresponding box should be small (*e.g.*, 1.0d-8) so that the reciprocal space portion of the Ewald sum is negligible. There must be **nbox** values.

ltwice: If true, the minimum image convention is applied twice. Necessary for the smaller (4x6; 20x26 Å) RPLC set-up. There must be **nbox** values.

lrplc: If true, there are special directions for SAFE-CBMC. Needed for ODS chains in RPLC simulations. There must be **nmolty** values.

5 Input/Output Files and Their Contents

5.1 Input Files

control.inc This file is located in the directory with the other MCCCS code files, so technically it is not an input file. In addition, like all **.inc** files, changes in **control.inc** do not take effect until the code is recompiled (see section 3). It is important, however, to check all parameters in **control.inc** and compile the code before running the simulation. Individual parameters are explained within **control.inc**. Array dimensions are also set in **control.inc**; thus, an "array out of bounds" error may require changing the array dimensions and recompiling the code.

fort.4 As detailed above, this is the input file that determines of what the system is comprised, how long to run the simulation and what kinds of moves to attempt. An error from the **readdat** file typically indicates a problem with the **fort.4** input file. The **fort.4** file must be included in all simulations.

fort.77 This is the restart file and must be present for all simulations in which **linit** (in **fort.4**) is false. For a continuing simulation, the final configuration file, **config##.dat**, can be copied to **fort.77** to continue from where the previous simulation ended. The following list includes, in order, the information that is included in **fort.77**. Multiple values listed on a single line are separated by spaces, not commas or any other characters.

total number of cycles for which the simulation has run
max atomic displacements in the x, y and z directions
max translational values in the x, y and z directions for molecules of type 1 in box 1
max rotational values in the x, y and z directions for molecules of type 1 in box 1
max translational values in the x, y and z directions for molecules of type 2 in box 1
max rotational values in the x, y and z directions for molecules of type 2 in box 1
. . .
max translation values in the x, y and z directions for molecules of type **nmolty** in box 1
max rotation values in the x, y and z directions for molecules of type **nmolty** in box 1
. . .
max translation values in the x, y and z directions for molecules of type **nmolty** in box **nbox**
max rotation values in the x, y and z directions for molecules of type **nmolty** in box **nbox**
fluctuating charges for molecule types 1 through **nmolty** in box 1
. . .
fluctuating charges for molecule types 1 through **nmolty** in box **nbox**
max volume displacements for boxes 1 through **nbox**
x, y and z lengths of box 1
. . .
x, y and z lengths of box **nbox**
total number of molecules in the simulation (**nchain**)
number of types of molecules (**nmolty**)
number of units for each molecule type (**nmolty** values)
molecule type of each chain (**nchain** values)
box number of each chain (**nchain** values)
x, y and z coordinates and the charge, q, of molecule 1
x, y and z coordinates and the charge, q, of molecule 2
. . .
x, y and z coordinates and the charge, q, of molecule **nchain**

input_struc.xyz Only necessary if **lbranch** in **fort.4** is true for any molecule. If so, this is where the structure of the molecule is specified.

fort.7 Only necessary for expanded ensemble calculations. This file holds the current values of the Lennard-Jones parameters.

fort.23 Only necessary if a SAFE-CBMC simulation is being performed. This file contains the probability histograms used in that algorithm. To adapt the probabilities, the final histograms that are given in the **fort.21** output file should be copied to **fort.23** before continuing the simulation.

The next four files are input files for simulations with tabulated potentials.

fort.41: Tabulated vibrational potentials. List the number of vibrational potentials, the vibration number fromm suvibe.f, the number of points per angstrom and the tabulated potential starting at 0.0. Repeat the last three points for each vibrational potential. Separate the different potentials with a line consisting of '1000 1000'.

fort.42: Tabulated 1-3 nonbonded 'bending' potentials. List the number of bending potentials, the bend number fromm suvibe.f, the number of points per angstrom and the tabulated potential starting at 0.0. Repeat the last three points for each 1-3 potential. Separate the different potentials with a line consisting of '1000 1000'. The tabulated bending is calculated using distances, not angles, so the 1-3 interactions must be included (see **internal 1-4**).

fort.43: Tabulated van der Waals potentials. List the number of vdW potentials, the bead numbers from suijtab.f (e.g., 114 115), the number of points per angstrom and the tabulated potential starting at 0.0. Repeat the last three points for each interaction (e.g., list both 114 115 and 115 114). Separate the different potentials with a line consisting of '1000 1000'.

fort.44: Tabulated electrostatic potentials. List the number of electrostatic potentials, the bead numbers fromm suijtab.f (*e.g.*, 114 115), the number of points per angstrom and the tabulated potential starting at 0.0. Repeat the last three points for each interaction (e.g., list both 114 115 and 115 114). Separate the different potentials with a line consisting of '1000 1000.' Currently, this is designed to work with Greg Voth's tabulated force-matched/coarse-grained potential, so each potential in fort.44 is the same. The tabulated potentials are then multiplied by the partial atomic charges within the code.

fort.61, fort.62, fort.63 The vibration, bending and torsion libraries for the molecule builder. Experimental???

5.1.1 Retired Input Files

fort.25 Only for use in zeolite calculations, this file stores the Lennard-Jones force field.

fort.91 Only for use in zeolite calculations, this file stores the tabulated zeolite potential energy.

5.2 Output Files

Note that the two characters included in the names of many of the output files (denoted here as ##) are the **run_num** and **suff** from the **fort.4** input file.

run##.dat: The majority of the information about the simulation including the fraction of accepted moves and the calculated energies and densities is written to **run##.dat**.

config##.dat The final configuration of the system as well as the maximum displacements and box lengths are contained in this file, which is also known as the restart file. Its format is identical to the format of the **fort.77** input file. To continue a simulation, this file should be copied to **fort.77**.

movie##.dat The movie file which holds the configurations of the system as output every **iblock**? **imov**? cycles. Used to determine the radial distribution functions, $g(r)$ among other things. Formerly called **fort.10**.

fort.11 The **isolute** movie file output every **isolute** cycles. Redundant to **movie##.dat** when special outputting is desired???

fort.12 Contains the box lengths, total energy and number of each molecule type for each box, output every cycle. Useful for many things.

fort.13 If there is a non-cubic, non-rectangular box in the simulation, this file will contain the angles between cell vectors, written out every cycle.

box#config##.xyz Lists the xyz coordinates and identifies the molecule type of each molecule in the system. In a format for VMD to visualize.

cell_param##.dat ???

fort.14, fort.15, fort.16, fort.17, fort.18 If the simulation is of a single molecule type and **idiele** is less than **nstep**, then these 5 files will contain averages related to the dielectric constant. **fort.14** and **fort.15** will have

$$\frac{4\pi}{3Vk_{\mathrm{B}}T}\frac{1}{4\pi\epsilon_0}\langle\mathbf{M}^2\rangle \quad \text{and} \quad \frac{4\pi}{3Vk_{\mathrm{B}}T}\frac{1}{4\pi\epsilon_0}(\langle\mathbf{M}^2\rangle - \langle\mathbf{M}_x\rangle^2 - \langle\mathbf{M}_y\rangle^2 - \langle\mathbf{M}_z\rangle^2)$$

respectively, where $\mathbf{M}$ is the total dipole moment of the box and $\mathbf{M}_x$, $\mathbf{M}_y$, and $\mathbf{M}_z$ its x, y and z components. Both quantities are unitless. Note that Allen & Tildesley relate the dielectric constant of the system ϵ, to these quantities *plus* 1. **fort.16, fort.17**, and **fort.18** will contain $\langle\mathbf{M}_x\rangle, \langle\mathbf{M}_y\rangle$, and $\langle\mathbf{M}_z\rangle$, respectively, in units of e Å.

fort.14, fort.15, fort.16, fort.17, fort.18, fort.19 If the simulation is of a single molecule type, **idiele** is less than **nstep** and **lnpt** is true, then these 6 files will contain the following averages: $\langle V\rangle, \langle V^2\rangle, \langle U\rangle, \langle U^2\rangle, \langle V\rangle\langle U\rangle$, and $\langle VU\rangle - \langle V\rangle\langle U\rangle$ where V is the box volume and U is the total energy, for each box. This is in addition to the dipole moment information contained in **fort.14-18** (see above).

fort.21 The final SAFE-CBMC probability histograms are contained in this file, if the SAFE-CBMC algorithm is used. Can be used to start a simulation with the new histograms by copying to **fort.23**.

fort.27: For dielectric constants. For an NVT simulation with ldielect=.true., fort.27 contains the x, y, and z components of the system dipole moment in units of $e\mathring{A}$. The quantities may be used to calculate the dielectric constant according to

$$\epsilon = 1 + \frac{1}{4\pi\epsilon_0}\frac{4\pi}{3VT}\left(\langle\mu^2\rangle - \langle\mu_x^2\rangle\langle\mu_y^2\rangle - \langle\mu_z^2\rangle\right) \tag{1}$$

(See Allen and Tildesley, page 161.)

fort.31, fort.32, fort.33 If the AVBMC algorithm is used, these files will contain the cluster statistics and energy ratios, which are output every cycle.

fort.55, fort.56 If **iheatcapacity** in **fort.4** is less than **nstep** and **lnpt**, which is in **control.inc**, is false, then **fort.55** contains the following averages: $< E >$ and $< E^2 >$. If **iheatcapacity** in **fort.4** is less than **nstep** and **lnpt**, which is in **control.inc**, is true, then **fort.56** contains the following averages: $< H >$ and $< H^2 >$.

save-config Like the **final-config** restart file but output every **nstep**/10 if **nstep** is greater than 100 MC cycles otherwise it is not written. Useful if a simulation ends unexpectedly so

that progress up to that point can be kept, in which case it should be copied to the next input configuration file, **fort.77**.

end2end_box1, end2end_box2, end2end_box3 These files contain the end to end vector distribution for box 1, box 2 and box 3 respectively. If the box is not present in simulation then that particular file will be empty.

rhoz_box1, rhoz_box2, rhoz_box3 These files contain the z density profiles for the respective boxes.

comrhoz_box1, comrhoz_box2, comrhoz_box3 These files contain center of mass z density profile for the respective boxes.

beadrdf_box1, beadrdf_box2, beadrdf_box3 These files contain bead-bead readial distribution functions for the the respective boxes.

comrdf_box1, comrdf_box2, comrdf_box3 These files contain center of mass readial distribution functions for the the respective boxes.

beadnum_box1, beadnum_box2, beadnum_box3 These files contain bead-bead number integrals for the the respective boxes.

comnum_box1, comnum_box2, comnum_box3 These files contain center of mass number integrals for the the respective boxes.

bendang_dist_box1, bendang_dist_box2, bendang_dist_box3 These files contain bending angle distribution for the respective boxes.

tors_frac_box1, tors_frac_box2, tors_frac_box3 These files contain torsion fractions for the respective boxes.

torsprob_box1, torsprob_box2, torsprob_box3 These files contain torsion angle probabilty distribution vs number of defects per chain, for the respective boxes.

Gauchedefects_box1, Gauchedefects_box2, Gauchedefects_box3 These files contain fraction gauche defects versus torsion for the respective boxes.

pattern_box1, pattern_box2, pattern_box3 These files contain pattern of g+, trans, g- for the respective boxes. Transform first number into base 3 to get the pattern of gauche defects where -1=g+, 0 = trans, 1=g+.

decoder This file contains decoder information for the gauche defect pattern.

Nrandomtest.dat Contains 10 numbers generated by the random number generator.

5.3 Notes on file management

Here are some basic guidelines about file management. To prevent output files from being overwritten, change the **run_num** and/or the **suff** in **fort.4**. Note that output files whose titles do not contain **run_num** and **suff** will be overwritten during each new run. Copy them to other files to avoid this.

5.3.1 Force field development

If one is developing a force field for a particular molecule then it is important to keep the output file (for example
% topmon > prod
where prod is the output file). It is not necessary to keep all other fort.* output files but for **final-config** , **fort.77**, **fort.4** to start the new trial. If you believe that you have parameters that are close to the desired value then you should carry out each simulation in a separate directory say T1, T2 (trial 1, trial2) etc and keep all the files for future reference. It is also a good idea to have a README file that has some basic iformation as to what is there in that particular directory.

5.3.2 Application oriented simulations

If the objective of the simulation is to to provide microscopic understanding of the system then you should **save all the files** of your simulation. You might want to create new directory for different simulation runs and copy the old files to files with names appended by date or any other way you want to identify them.

6 Example Simulation

6.1 Initialization, Melting and Cooling

Let's consider the calculation of the vapor-liquid coexistence curve for methanol. To do this, we'll use the NVT-Gibbs ensemble to determine the saturated liquid and vapor densities for a range of subcritical temperatures. Determining the vapor pressures will also allow us to predict the normal boiling point.

After specifying the connectivity and force field parameters in a **fort.4** file, we want to run the simulation. Since we don't have a restart file (**fort.77**), we set **linit** to true in **fort.4**. In this case, the box lengths for both simulation boxes will be read from **fort.4**, and you should pick densities that correspond roughly to a liquid and a vapor. Place between 10 and 33% of the molecules in the vapor box. If we're simulating 300 molecules (**nchain** = 300) then put 50 in the vapor phase and use 29 and 38 Å as box lengths for the liquid and vapor phases, which correspond to 0.55 and 0.048 g cm^{-3}, respectively. Split up the attempted Monte Carlo move probabilities so that we perform *no* volume moves (**pmvol** = 0) and evenly divide the probability into CBMC, translation and rotation by setting **pmcb** = 0.33 and **pmtra** = 0.67 and leaving all other probabilities at zero. Since we want to "melt" this structure away from the lattice it was set up on, set the temperature high, say 10,000 K, and run for 2,000 cycles to create a disordered system. For this process, a potential cutoff of 9 Å is sufficient.

Once this is done, we don't want to initialize again, so set **linit** to false and copy **final-config** to **fort.77**. At this point, we want to "cool" the simulation, so we leave the cutoff and

probabilities the same and change the temperature to around 90% of the critical temperature, say 475 K, and then run for at least 5,000 cycles. If you set **iblock** to 1,000, at the end of the output file you will see 5 values of the energy (among other things) averaged over 1,000 cycles each, and can get a feel for how much it's changing. It'll probably be fairly steady after 2-3,000 cycles.

6.2 Equilibration

Now that we've brought the system down to a temperature that we want to get an accurate value of the saturated densites for, the next step is equilibration. Since we want to equilibrate density and chemical potential, we need to have moves that will change these. They are the volume and swap moves, respectively. We also want a more accurate calculation of the potential energy, so set the cutoff (**rcut**) to 14 Å. Don't forget to copy **final-config** to **fort.77** for each new simulation.

Ideally we want one accepted volume move for every ten cycles, so we should set **pmvol** to 0.2 / $N_{\text{molecules}}$ as we will be going for a target acceptance ratio of 50% for volume moves (**tavol** = 0.5). Set **iratv** so that the maximum volume displacement is allowed to change, say once every 250 cycles. Now we want to allow some swap moves too, so set **pmswap** = 0.01 and run the simulation for 10,000 cycles. Like the volume move, we want one accepted every 10 cycles, but we have no target acceptance ratio, so we have to see by trial and error what value of **pmswap** is appropriate.

For the stated probabilities, here's a sample of the output file as it pertains to the swap move:

Molecule swap

molecule typ = 1
between box 1 and 2 into box 1
uattempts = 14021.0 attempts = 14021.0 accepted = 254.0
suc.growth % = 99.501 accepted % = 1.812
between box 1 and 2 into box 2
uattempts = 13851.0 attempts = 13851.0 accepted = 275.0
suc.growth % =100.000 accepted % = 1.985

As we ran 10,000 cycles, that means we had $10000/(254 + 275) \approx 19$ cycles between swap moves. Since we want that to be close to 10, we can increase **pmswap** to $0.01 \times \frac{19}{10} = 0.019$ for the next simulation. Note that since there is an overall flux of molecules from the liquid to the vapor box, the simulation has not reached equilibrium.

6.3 Production

Once we have reached stable vapor and liquid phases in our simulation, it's time to start producing data to collect. At this point, we should know that we'll be getting one accepted volume and one accepted swap move every 10 cycles. We don't need to update the maximum displacements, so we can set **iratio** and **iratv** to a value greater than or equal to **nstep**. These two values must not change during the production period; allowing them to change would violate the principle of microscopic reversibility. In fact, you could set the maximum x, y and z translational displacements to the average of the three, same for rotational displacements, in the **fort.77** file, if you want.

We can start calculating the pressure more frequently now, but it's still a fairly expensive calculation, so set **iratp** to say 5, which will calculate the pressure every 5 cycles. You will see that the pressure of a liquid is noisy, very noisy, and the standard deviation can be larger than the actual value! So in a case like this (NVT-Gibbs ensemble) we say that the pressure between the two boxes is equilibrated due to the volume move, and just read the pressure from the vapor box.

6.4 Simulations at Lower Temperatures

When we have a decent result for 475 K, we can start thinking about calculating another state point at a lower temperature. To map out the coexistence curve you might go down in 50 K increments to 275 K. You could go lower but low temperatures take longer to equilibrate.

To keep the files separate, make a new directory for each temperature, in this case 425 K. Copy in the **fort.4** and **fort.77** files, and change the temperature in **fort.4** to 425. We will again have to equilibrate our simulation before we start collecting data. This mainly means volume and swap moves, as the vapor (and to a lesser extent liquid) density generally has a strong temperature dependence, especially near the critical point. Also, if you consider the lever rule, the relative amount of molecules in the vapor and liquid boxes will change (this is the same phenomena, just expressed differently). This means if we hold our box sizes equal, when we lower the temperature molecules will leave the vapor box in favor of the liquid. Our number of accepted swap moves will also decrease, as it becomes harder to swap at lower temperatures, so that we should adjust **pmswap** to bring the number back up to one accepted move per 10 cycles.

To ensure that we keep around 50 molecules in the vapor phase, we will have to *rescale* the vapor box lengths, but by how much? The answer is to compare the current number of molecules in the vapor phase to our target amount. Say that after 10,000 cycles at 425 K we wind up with 12 molecules in the vapor phase. We have $\frac{50}{12} \approx 4$ times too few molecules, so we want to increase the volume by a factor of 4. Since $V = L^3$, we need to uniformly increase each box length (L) by $4^{1/3}$ or 1.587. With our desired box length known, we then need to edit the restart file, **fort.77** to reflect this. For this case (one molecule type, two boxes), the lengths of the two boxes are on the 9^{th} and 10^{th} lines of the **fort.77** file. Note that changing the values in **fort.4** will have no effect on the simulation! Now equilibrate again and we should have close to 50 molecules in the vapor phase. From here, you can then

start a production run to collect data at 425 K, and repeat the process for 375 K, 325 K, and so on. At some point, the amount of accepted swaps will become so low that it won't be possible to have one per 10 cycles. Then you can increase the number of trial positions investigated by increasing **nchoi1** and **nchoi**, however at increased computational cost. At 275 K, we might have to live with, say, only one accepted per 20 cycles.

7 MCCCS files

This section includes a list of the files included in the MCCCS directory.

... ...

8 Suijtab.f

suijtab.f is a file located in the MCCCS directory that contains parameters for each Lennard-Jones site. This information is transferred to **fort.4** via an identification number

The following information is included for each site.

- **sigi:** The L-J σ parameter in Å.
- **epsi:** The L-J ϵ parameter in K.
- **mass:** The total molecular mass of the site.
- **qelect:** The partial charge in e. Not included if neutral.
- **lij:** A logical variable Not included if neutral.
- **lqchg:** A logical variable Not included if neutral.
- **chname:** A descriptive name for the site.
- **chemid:** A single-letter description. Used when visualizing the system in VMD.

To add a new site to the code, please use a number greater than 400 and email the updated **suijtab.f** file to the entire group.

9 Thermodynamic integration in stages

See Maginn's paper for the details of this method. In Topmon, here are the changes. There is a new logical variable called lmipsw in control.inc. If true, it does the thermodynamic integration (either use NVT or use NPT with zero percant volume move - hence NVT). Will fail for any other ensemble. And, only checked for one component systems.

No change in fort.4 - the input file is unaffected by the thermodynamic integration part. However, using lmipsw requires one new file called fort.35. The file is as follows -
1. **lwell** - set to true if stages b and c are done, false in case of stage a. For a definition of stages, see Maginn's paper.
2. **awell(i,j)** - input the strength of the external well in K in a matrix notation (interaction of i molecule site with j lattice site).
3. **bwell** - parameter for gaussian well width - 0.5 might be a good number.
4. **lstagea**, **lstageb**, or **lstagec** - make one of them true.
5. **etais**, **lamdbais** - the final weak potential, and value of **lambda** for that stage (see the paper).
6. box length - specify the boxlength at the beginning of stagea and the end of stagec (full stages) in hmatrix notation if lsolid is true and lrect is false. If lsolid is false, use the box length of the two cubic boxes. Cannot mix cubic/noncubic boxes - best is to use hmatrix notation.
7. **iratipsw** - the number of cycles in which the integrand is updated. If lstageb is true, make sure that this is an integral multiple of iratp.
8. reduced coordinates of all the lattice sites.

10 New expanded ensemble

It does work for changing the identity of one molecule (e.g., from ideal to a full molecule). e.g., for growing a solute molecule in solvent in a one-box NPT or NVT ensemble. It may work for Gibbs (swapping that molecule in one of its identities to the other box and then growing it) - but not extensively tested recently. This part of the code requires one change to the input in fort.4 - a variable called pmexpc1 - the probabiliity of doing the new expanded ensemble move. Other than that, there is a variable called lexpee in control.inc. If this logical variable is true, then expanded ensmble is done and another input file called fort.45 is required. Also there is a smax variable - maximum number of stages that intialize the arrays.

The input file is as follows -
1. **imolty** on which ee is performed - molecule type. For example imolty = 1 is for the solvent, imolty = 2 is for the solute particles already in the solution, and imolty = 3 is the solute particle that we will try to grow in the solution for this simulation.
2. **nmolty1**: number of actual types of molecules in the simulation. In the above example - it is 3 (1 solvent, 1 solute molecules in full form, and 1 solute molecule that will go through stages).
3. final state: **fmstate** - the final state of the growing molecule (when it is fully grown). This determines the number of molecule types that are needed in fort.4. In the above example, it is fmstate+2. Each fmstate stages have one molecule type each, and one for the solvent, and one for the already present solutes.
4. weights associated with each stage: fmstate values. Could start with all zero and then optimize it so that all stages are visited with reasonable frequency.
5. **sstate1** and **sstate2**: In case expanded Gibbs ensemble is performed, the interdiate stages

when the growing molecule is transferred from one box (sstate1) to the other (sstate2). Should be consequitive numbers. In case NPT or NVT expanded ensemble - make these numbers greater than fmstate (need to adjust smax in control such that these numbers are less than smax).
6. **eeratio**: In case of expanded Gibbs ensemble, the probability of exchanging the tagged particle (that is undergoing ee) at its end stages (full molecules in either box) with an untagged solute particle. For example, if stage 1 is in box 1 and stage 10 in box 2 - and at both these stages the tagged molecule is full molecule (just in different boxes) - then there should be a way to make another solute molecule tagged (otherwise only one molecule will switch between boxes). In case of NPT or NVT EE, make eeratio to be less than 0.
7. **mstate**: the current stage of the tagged molecule.

11 Additions for dipole moment and electric field

dipole.dat For an NVT simulation with **ldielect** equal to true, dipole.dat contains the x, y, and z components of the dipole moment (μ_x, μ_y, and μ_z) in units of e Å. These quantities may be used to calculate the dielectric constant using the following equation:

$$\epsilon = 1 + \frac{1}{4\pi\epsilon_0}\frac{4\pi}{3VT}\left(\langle\mu^2\rangle - \langle\mu_x\rangle^2 - \langle\mu_y\rangle^2 - \langle\mu_z\rangle^2\right) \tag{2}$$

see Allen and Tildesley, page 161.

lelect_field must be set to true in order for the electric field calculation to be turned on.

In fort.4:

Elect_field: the electric field strength, in units of V/Å. The electric field is applied in the z-direction. (this appears in fort.4 after fqtemp)

In order to calculate the interaction with an electric field, a new function (exfield) is called in sumup, energy, and boltz. exfield.f calculates the interaction with an electric field, given by:

$$\begin{aligned} U_{\text{field}} &= -\mu \cdot E \\ &= -q * r_z * E \end{aligned}$$