SAND2018-7268C

LAMMPS Tutorial

Stan Moore

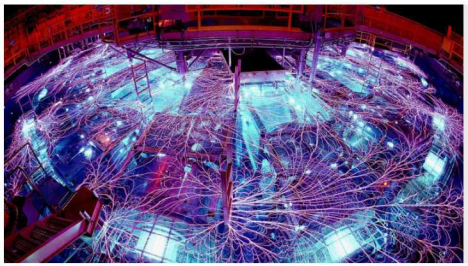
Modeling Supra-molecular Structures with LAMMPS

Philadelphia, PA

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.. SAND NO. 2011-XXXXP

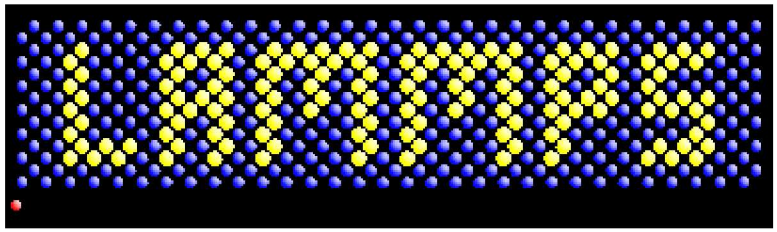
About Me

- Stan Moore
  - Core LAMMPS developer at Sandia National Labs in Albuquerque, New Mexico
  - Been at Sandia for 6 years
  - Main developer of the KOKKOS package in LAMMPS (runs on GPUs and multi-core CPUs)
  - Expertise in long-range electrostatics
  - PhD in Chemical Engineering, dissertation on molecular dynamics method development for predicting chemical potential



Tutorial Overview

1. LAMMPS theory and methods
2. Download, build, and run LAMMPS
3. Work through two examples: Lennard Jones liquid and rhodopsin (solvated protein)



Getting Help

Sandia National Laboratories

- Look at documentation that came with your version of LAMMPS (latest version here: http://lammps.sandia.gov/doc/Manual.html)
- Search mail list archives here: https://sourceforge.net/p/lammps/mailman/lammps-users
- Subscribe to the LAMMPS mail list here: http://lammps.sandia.gov/mail.html and then post questions
- Look at mail list posting guidelines first: http://lammps.sandia.gov/guidelines.html

LAMMPS: Modular and Flexible

Sandia National Laboratories

- LAMMPS is easily extendable (don't need to modify core source code to add a diagnostic)
- Several optional packages can be installed
  - "USER-" packages are maintained by non-core LAMMPS developers
- Many optional diagnostics and commands can be used
  - Computes: only give information about the simulation: e.g. temperature
  - Fixes: modify atom properties such as forces: e.g. thermostat
- LAMMPS is scalable to large number of processors using the MPI domain decomposition method

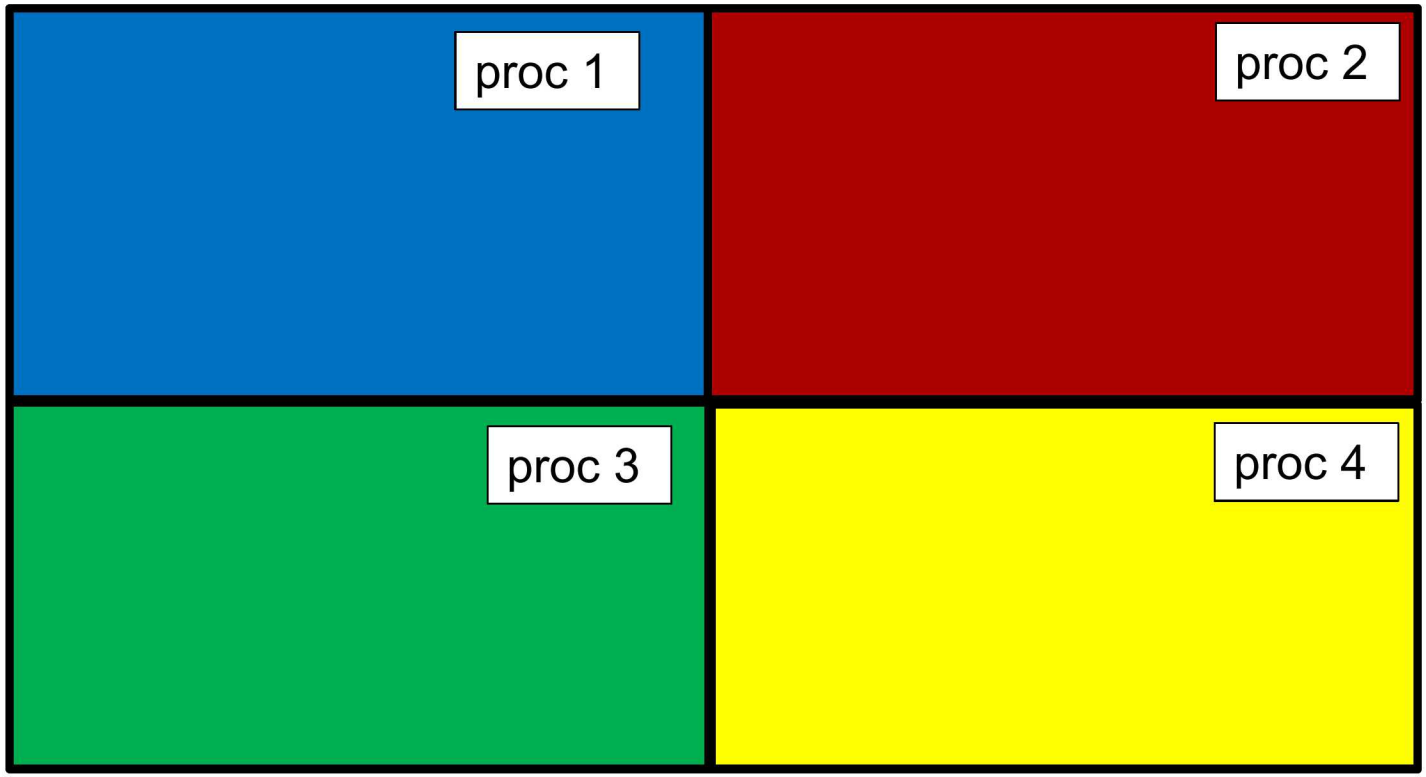
LAMMPS Code Design: C++

Sandia National Laboratories

- Why c++?
  - Allows object-oriented design, encapsulation
  - Allows virtual inheritance, which reduces code duplication
  - Performance still on par with older Fortran version of LAMMPS
- What style of c++?
  - Object-oriented c
  - Use c for low-level operations and c++ for high level (e.g. classes with virtual inheritance)
  - In LAMMPS core, simple data structures, no stl vectors, few templates, etc.
  - Multidimensional arrays are contiguous in memory
  - In USER packages, more freedom (can use stl vectors, templates, etc.)

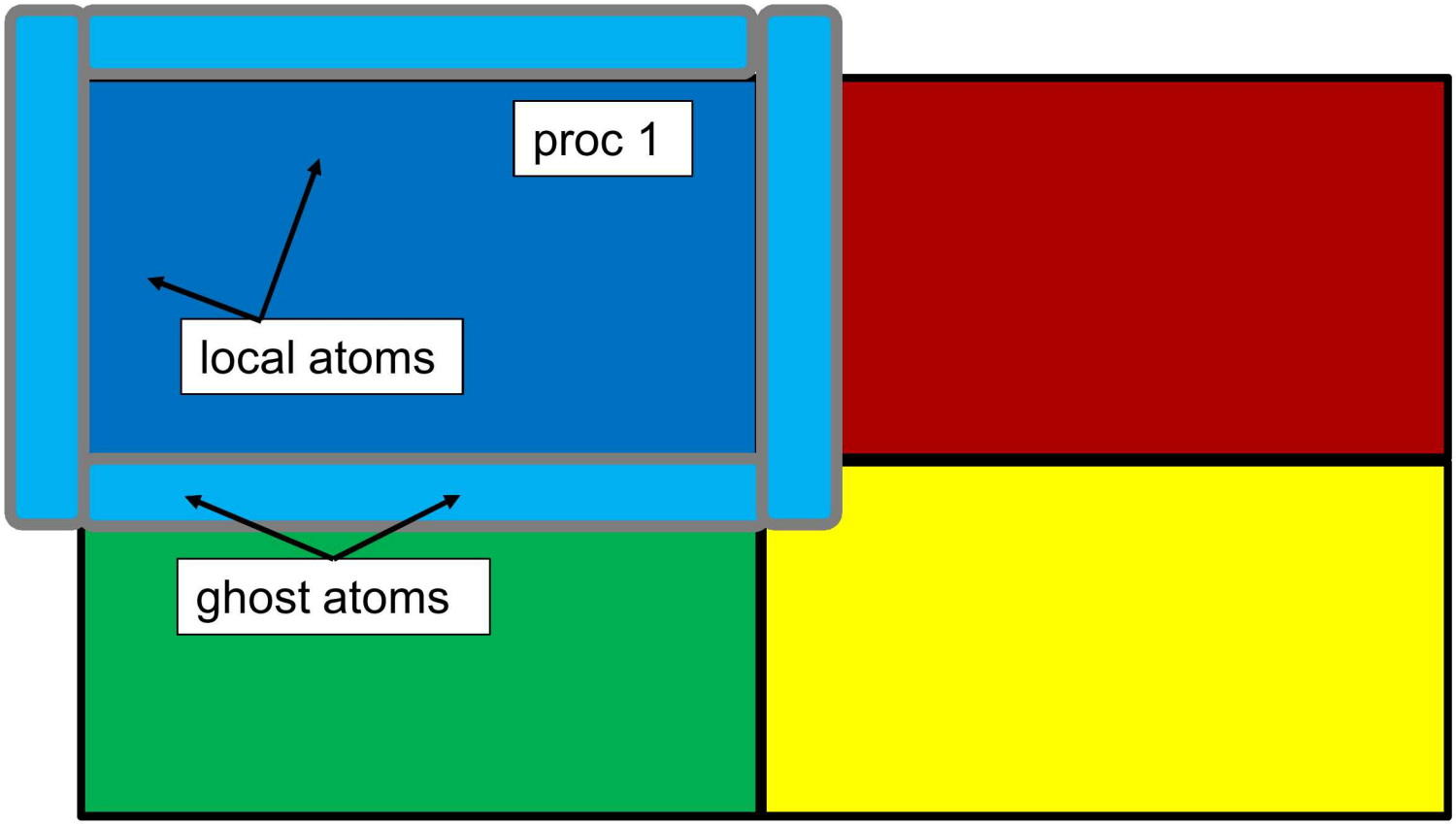
MPI Parallelization Approach

- Domain decomposition: each processor owns a portion of the simulation domain and atoms therein



Ghost Atoms

- The processor domain is also extended include needed ghost atoms (copies of atoms located on other processors)



Communication Patterns

Sandia National Laboratories

- *Forward* communication updates ghost atoms properties (such as positions) from the corresponding real atoms on a different processor
- *Reverse* communication takes properties (such as forces) accumulated on ghost atoms and updates them on the corresponding real atoms on a different processor
- *Exchange* communication migrates real atoms from one processor to another
- *Border* communication creates new ghost atoms
- LAMMPS tries to minimize the number of MPI calls required between subdomains

Strong vs Weak Scaling

Sandia National Laboratories

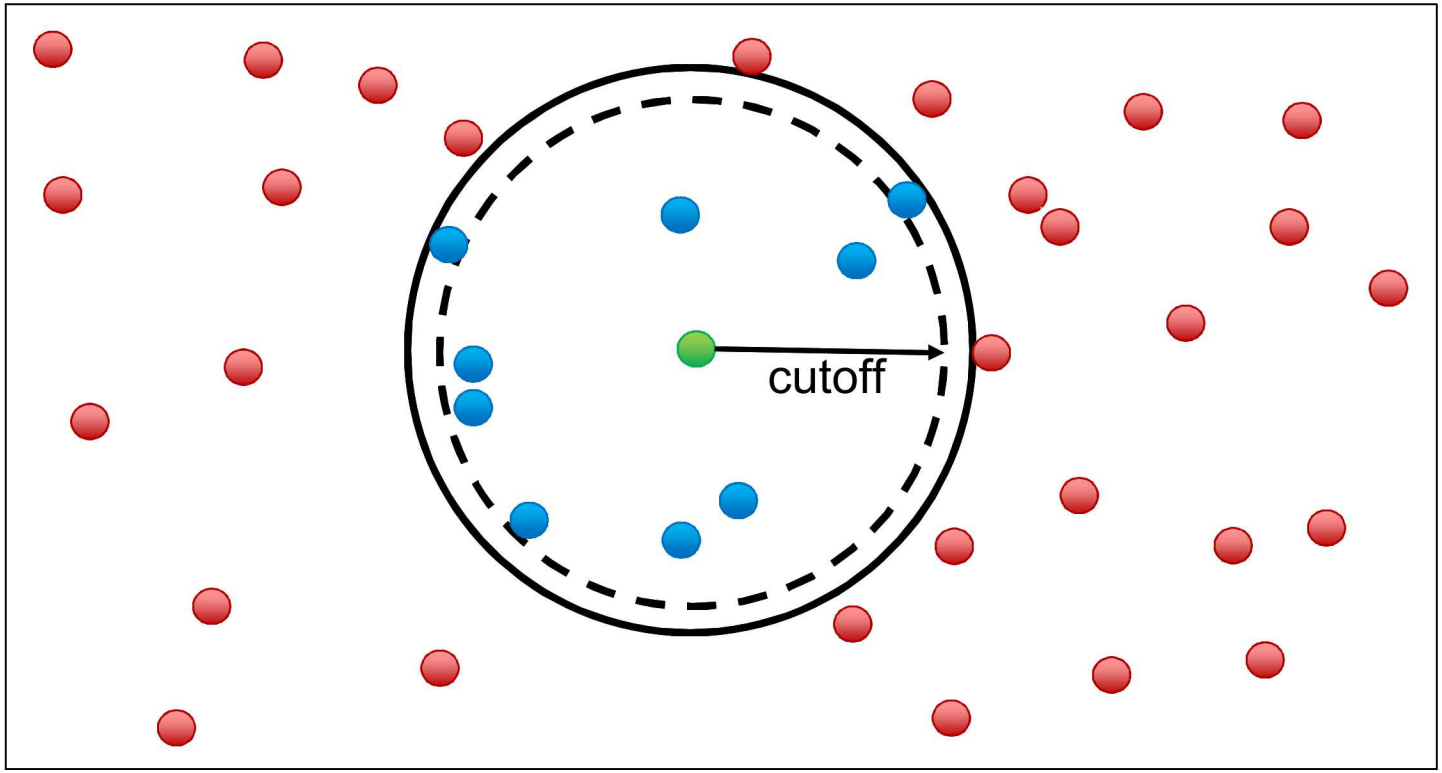
- Strong scaling: hold system size fixed while increasing processor count (# of atoms/processor decreases)
- Weak scaling: increase system size in proportion to increasing processor count (# of atoms/processor remains constant)
- For perfect strong scaling, doubling the processor count cuts the simulation time in half
- For perfect weak scaling, the simulation time stays exactly the same when doubling the processor count
- Harder to maintain parallel efficiency with strong scaling because the compute time decreases relative to the communication time

Atom IDs

- atom id is a global property stored in atom->tag variable
- *Ntotal* is the total number of atoms in the simulation
- *Nlocal* is the number of atoms owned by each MPI rank
- Local atom IDs run from 0 … *Nlocal* on each processor
- Global atom IDs run from 0 … *Ntotal* across processors
- There can be multiple ghost atoms with the same id for small systems with long cutoffs => no minimum image convention limitation

Neighbor Lists

- Neighbor lists are a list of neighboring atoms within the interaction cutoff + skin for each central atom
- Extra skin allows lists to be built less often



Neighbor Options

- *delay* setting means never build new lists until at least N steps after the previous build
- *every* setting means build lists every M steps (after the delay has passed)
- If the *check* setting is *no*, the lists are built on the first step that satisfies the *delay* and *every* settings. If the *check* setting is *yes*, an actual build only occurs if some atom has moved more than half the skin distance (still respects *delay* and *every)*
- *exclude* turns off pairwise interactions between certain pairs of atoms, by not including them in the neighbor list

```
neighbor 0.3 bin
neigh_modify every 2 delay 10 check yes
neigh_modify exclude type 2 3
```

Newton Option

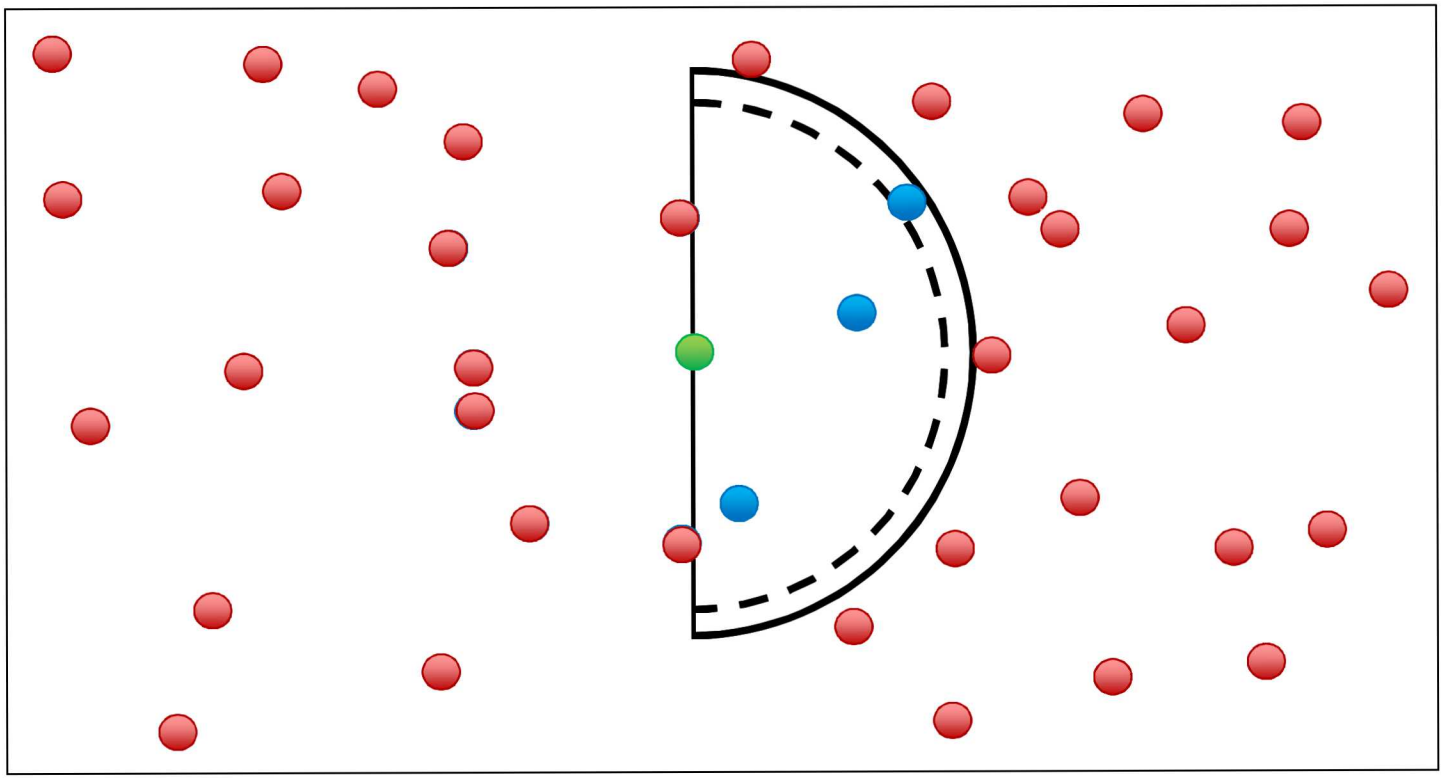
Sandia National Laboratories

- Setting the newton flag to *off* means that if two interacting atoms are on different processors, both processors compute their interaction and the resulting force information is not communicated
- Setting the newton flag to *on* means a modest savings in computation at the cost of two times more communication
- Performance depends on problem size, force cutoff lengths, a machine's compute/communication ratio, and how many processors are being used

```
newton on #default
newton off
```

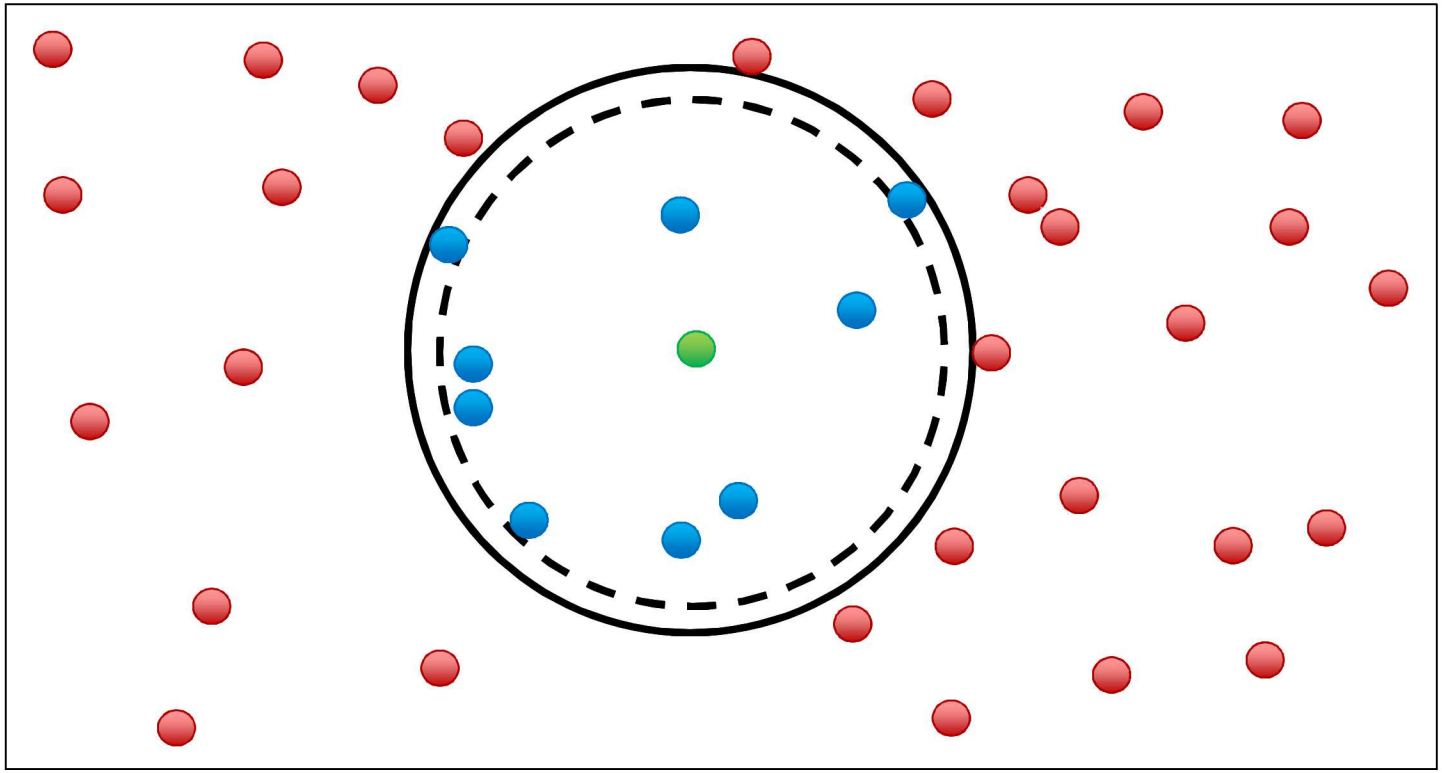
Half Neighbor List

- With newton flag on, each pair is stored only once (usually better for CPUs)



Full Neighbor List

- Each pair stored twice which doubles computation but reduces communication (can be faster on GPUs)

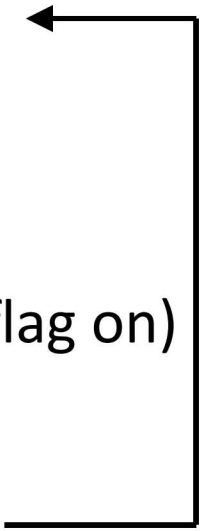


Basic MD Timestep

Sandia National Laboratories

- During each timestep (without neighborlist build):

1. initial integrate
2. MPI forward communication
3. compute forces (pair, bonds, etc.)
4. MPI reverse communication (if newton flag on)
5. final integrate
6. output (if requested on this timestep)



*Computation of diagnostics (fixes or computes) can be scattered throughout the timestep

Pair Styles

- In LAMMPS, force fields are called “pair styles”
- Need to specify:
  - style
  - cutoff distance
  - interaction coefficients
  - … and maybe more, depending on the pair style

```
pair_style      lj/cut 2.5
pair_coeff      1 1 1.0 1.0 2.5
```

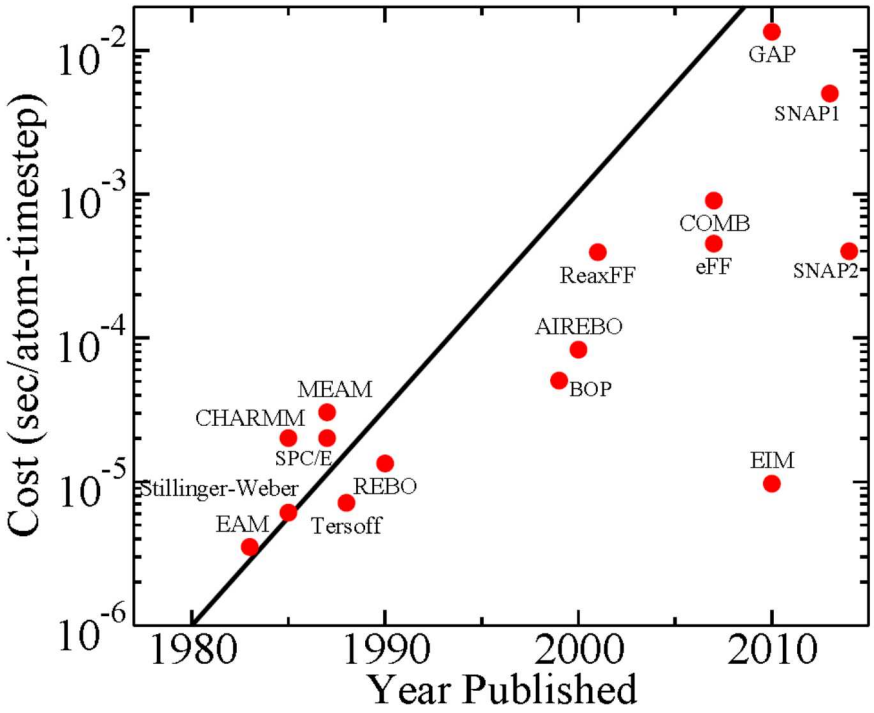
LAMMPS Potentials

Sandia National Laboratories

- See lammps.sandia.gov/bench.html#potentials

Potential	System	Atoms	Timestep	CPU	LJ Ratio
Granular	chute flow	32000	0.0001 tau	5.08e-7	0.34x
FENE bead/spring	polymer melt	32000	0.012 tau	5.32e-7	0.36x
Lennard-Jones	LJ liquid	32000	0.005 tau	1.48e-6	1.0x
DPD	pure solvent	32000	0.04 tau	2.16e-6	1.46x
EAM	bulk Cu	32000	5 fmsec	3.59e-6	2.4x
Tersoff	bulk Si	32000	1 fmsec	6.01e-6	4.1x
Stillinger-Weber	bulk Si	32000	1 fmsec	6.10e-6	4.1x
EIM	crystalline NaCl	32000	0.5 fmsec	9.69e-6	6.5x
SPC/E	liquid water	36000	2 fmsec	1.43e-5	9.7x
CHARMM + PPPM	solvated protein	32000	2 fmsec	2.01e-5	13.6x
MEAM	bulk Ni	32000	5 fmsec	2.31e-5	15.6x
Peridynamics	glass fracture	32000	22.2 nsec	2.42e-5	16.4x
Gay-Berne	ellipsoid mixture	32768	0.002 tau	4.09e-5	28.3x
AIREBO	polyethylene	32640	0.5 fmsec	8.09e-5	54.7x
COMB	crystalline SiO2	32400	0.2 fmsec	4.19e-4	284x
eFF	H plasma	32000	0.001 fmsec	4.52e-4	306x
ReaxFF	PETN crystal	16240	0.1 fmsec	4.99e-4	337x
ReaxFF/C	PETN crystal	32480	0.1 fmsec	2.73e-4	185x
VASP/small	water	192/512	0.3 fmsec	26.2	17.7e6
VASP/medium	CO2	192/1024	0.8 fmsec	252	170e6
VASP/large	Xe	432/3456	2.0 fmsec	1344	908e6

Accuracy = Higher Cost



Moore's Law for Interatomic Potentials

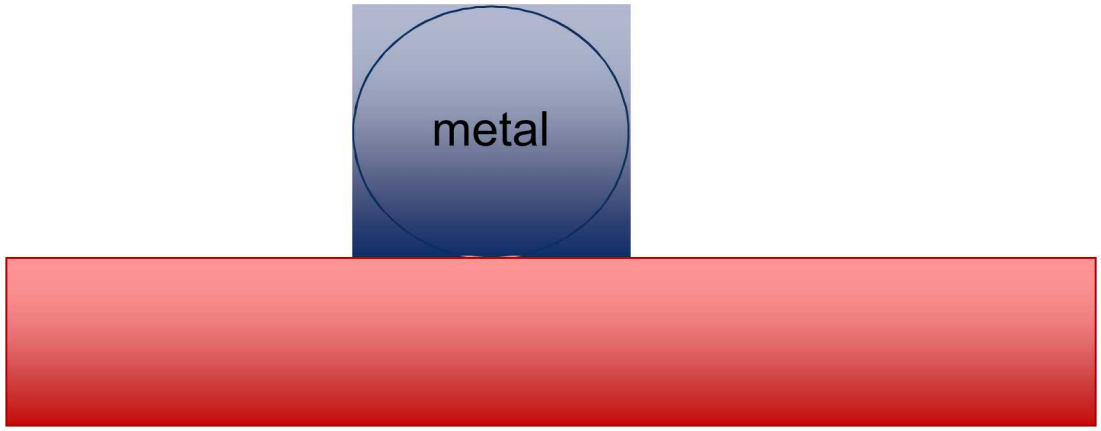
Plimpton and Thompson, MRS Bulletin (2012).

Hybrid force fields

- Pair hybrid: exactly one pair style is assigned to each pair of atom types (can be different pair styles for different atom types)
- Pair hybrid/overlay: one or more pair styles can be assigned to each pair of atom types
- The assignment of pair styles to type pairs is made via the pair_coeff command

Hybrid force fields—Example

- Metal droplet on an LJ surface
  - metal → metal atoms interact with *eam* potential
  - surface → surface atoms interact with *lj/cut* potential
  - metal/surface interaction is also computed via a *lj/cut* potential




```
pair_style hybrid lj/cut/coul/cut 10.0 eam lj/cut 5.0
```

Syntax specifying advanced force field input

Sandia National Laboratories

- Look in the LAMMPS documentation (i.e. http://lammps.sandia.gov/doc/pair_reax.html)
- Shows pair_style name
- Shows syntax with required and optional keywords
- Gives examples of command use
- Gives a description of the keywords
- Also gives restrictions, related commands, and default values of optional keywords

EAM, SW, and Tersoff Syntax

```
pair_style eam
pair_coeff * * cuu3
pair_coeff 1*3 1*3 niu3.eam

pair_style sw
pair_coeff * * si.sw Si
pair_coeff * * GaN.sw Ga N Ga

pair_style tersoff
pair_coeff * * Si.tersoff Si
pair_coeff * * SiC.tersoff Si C Si
```

- Can use forcefield files in /potentials folder

ReaxFF Syntax

Sandia National Laboratories

Docs » pair_style reax/c command

pair_style reax/c command

pair_style reax/c/kk command

Syntax

```
pair_style reax/c cfile keyword value
```

- cfile = NULL or name of a control file
- zero or more keyword/value pairs may be appended

```
keyword = checkqeq or lgvdw or safezone or mincap
  checkqeq value = yes or no = whether or not to require qeq/reax fix
  lgvdw value = yes or no = whether or not to use a low gradient vdW correction
  safezone = factor used for array allocation
  mincap = minimum size for array allocation
```

Examples

```
pair_style reax/c NULL
pair_style reax/c controlfile checkqeq no
pair_style reax/c NULL lgvdw yes
pair_style reax/c NULL safezone 1.6 mincap 100
pair_coeff * * ffield.reax C H O N
```

Atom-style Formats

Sandia National Laboratories

- Different force fields require different data to be stored with atoms (i.e. positions, charges, bonds, etc.)
- Atomic:
  - Basic style that can be used for simple potentials without bonds or charges
  - E.g. Lennard Jones fluid or metals
- Full:
  - More comprehensive style that includes charges and molecular topology (bonds, angles, dihedrals, and impropers)
  - E.g. water or polymers

```
atom_style    atomic
atom_style    full
```

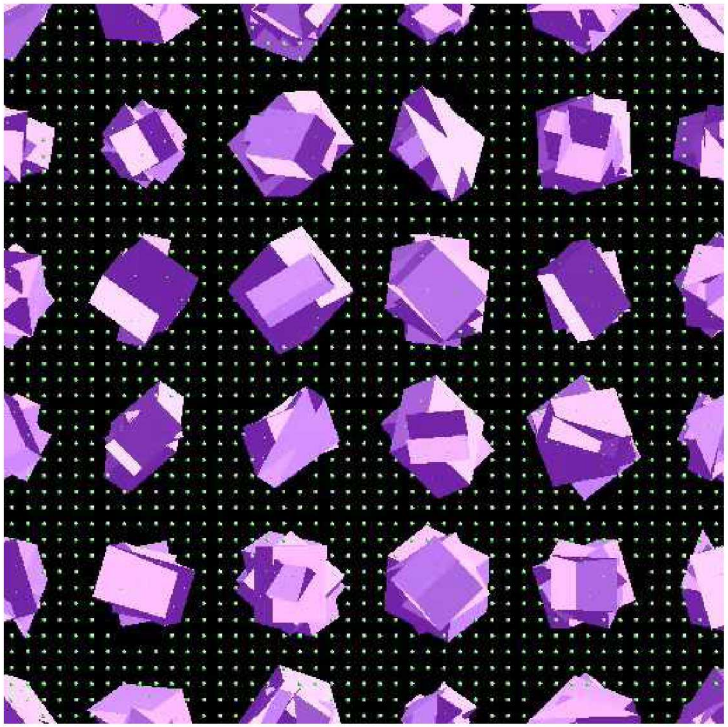
Atom-style Formats

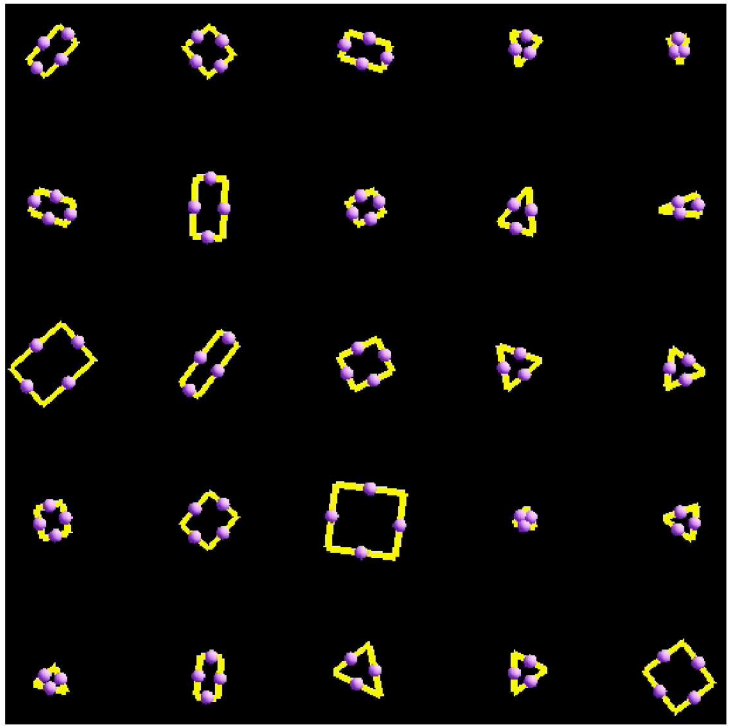
Sandia National Laboratories

angle	atom-ID molecule-ID atom-type x y z
atomic	atom-ID atom-type x y z
body	atom-ID atom-type bodyflag mass x y z
bond	atom-ID molecule-ID atom-type x y z
charge	atom-ID atom-type q x y z
dipole	atom-ID atom-type q x y z mux muy muz
dpd	atom-ID atom-type theta x y z
electron	atom-ID atom-type q spin eradius x y z
ellipsoid	atom-ID atom-type ellipsoidflag density x y z
full	atom-ID molecule-ID atom-type q x y z
line	atom-ID molecule-ID atom-type lineflag density x y z
meso	atom-ID atom-type rho e cv x y z
molecular	atom-ID molecule-ID atom-type x y z
peri	atom-ID atom-type volume density x y z
smd	atom-ID atom-type molecule volume mass kernel-radius contact-radius x y z
sphere	atom-ID atom-type diameter density x y z
template	atom-ID molecule-ID template-index template-atom atom-type x y z
tri	atom-ID molecule-ID atom-type triangleflag density x y z
wavepacket	atom-ID atom-type charge spin eradius etag cs_re cs_im x y z
hybrid	atom-ID atom-type x y z sub-style1 sub-style2 ...

Triangle and line particle examples

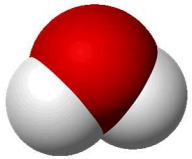
Sandia National Laboratories

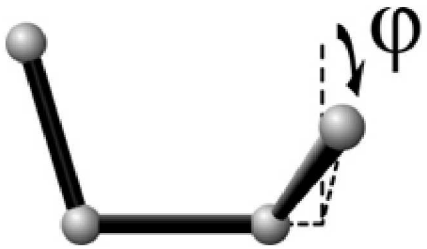




Molecular Topology

- Bonds: constrained length between two atoms
- Angles: constrained angle between three atoms
- Dihedrals: interactions between quadruplets of atoms
- Impropers: “improper” interactions between quadruplets of atoms

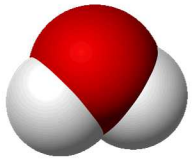





```
bond_style      harmonic
angle_style     charmm
dihedral_style  charmm
improper_style  harmonic
```

Fix Shake and Fix Rattle

- Applies bond and angle constraints to specified bonds and angles in the simulation by either the SHAKE or RATTLE algorithms
- Typically enables a longer timestep
- In LAMMPS, only small clusters of atoms can be constrained



```
fix     1 all shake 0.0001 5 0 m 1.0 a 232
```

Special Bonds

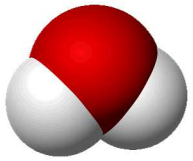
- Allow non-bonded LJ or Coulombic interactions between the pair of atoms to be excluded (or reduced by a weighting factor)
- Weighting factors are a scaling pre-factor on the energy and force between the pair of atoms. 1.0 means include the full interaction; 0.0 means exclude it completely
- Fist coefficient is the weighting factor on 1-2 atom pairs, which are pairs of atoms directly bonded to each other
- Second coefficient is the weighting factor on 1-3 atom pairs which are those separated by 2 bonds
- Third coefficient is the weighting factor on 1-4 atom pairs which are those separated by 3 bonds

```
special_bonds charmm
```

Long-Range Electrostatics

Sandia National Laboratories

- Truncation doesn't work well for charged systems due to long-ranged nature of Coulombic interactions
- Use Kspace style to add long-range electrostatics:
  - PPPM—usually fastest, uses FFTs
  - Ewald—potentially most accurate, but slow for large systems
  - MSM—multigrid method that also works for non-periodic systems
- Usually specify a relative accuracy (1e-4 or 1e-5 typically used)
- Use pair_style *coul/long such as lj/cut/coul/long instead of *coul/cut



```
pair_style      lj/cut/coul/long 10.0
kspace_style    pppm 1e-4
```

2D Slab Geometry with Kspace

- The *slab* keyword allows a Kspace solver to be used for a systems that are periodic in x,y but non-periodic in z
- Must use a boundary setting of "`boundary p p f`"
- Actually treats the system as if it were periodic in z, but inserts empty volume between atom slabs and removing dipole inter-slab interactions so that slab-slab interactions are effectively turned off
- May need to use reflecting walls in the z-dimension


```
boundary        p p f
kspace_modify   slab 3.0
```

Run vs Minimize

Sandia National Laboratories

- "run" command updates velocities and positions based on forces. System may blow up and crash LAMMPS if atoms overlap!
- "minimize" command minimizes energy of the system by iteratively adjusting atom coordinates
  - Good to minimize first if you built your system using an external tool
  - Prevents LAMMPS crashing from overlapping atoms
- Can use "run" after "minimize"

```
minimize    1.0e-4 1.0e-6 100 1000
run         100
```

LAMMPS Files

Sandia National Laboratories

- Input file: text file with LAMMPS commands used to run a simulation
- Log file: text file with thermodynamic output from simulation
- Dump file: snapshot of atom properties, i.e. atom forces
- Restart file: binary checkpoint file with data needed to restart simulation
- Data file: text file that can be used to start or restart simulation

LAMMPS Input File

- Uses custom powerful scripting language
- Can define variables
- Order of some commands matters
- Can use python
- Many commands take a custom name, and operate “all” atoms

```
compute myKe all ke/atom
fix myHisto all ave/histo 1 1 100 0.0 1.5 10
  c_myKe file temp.histo mode vector
```

Input Script Variables

Sandia National Laboratories

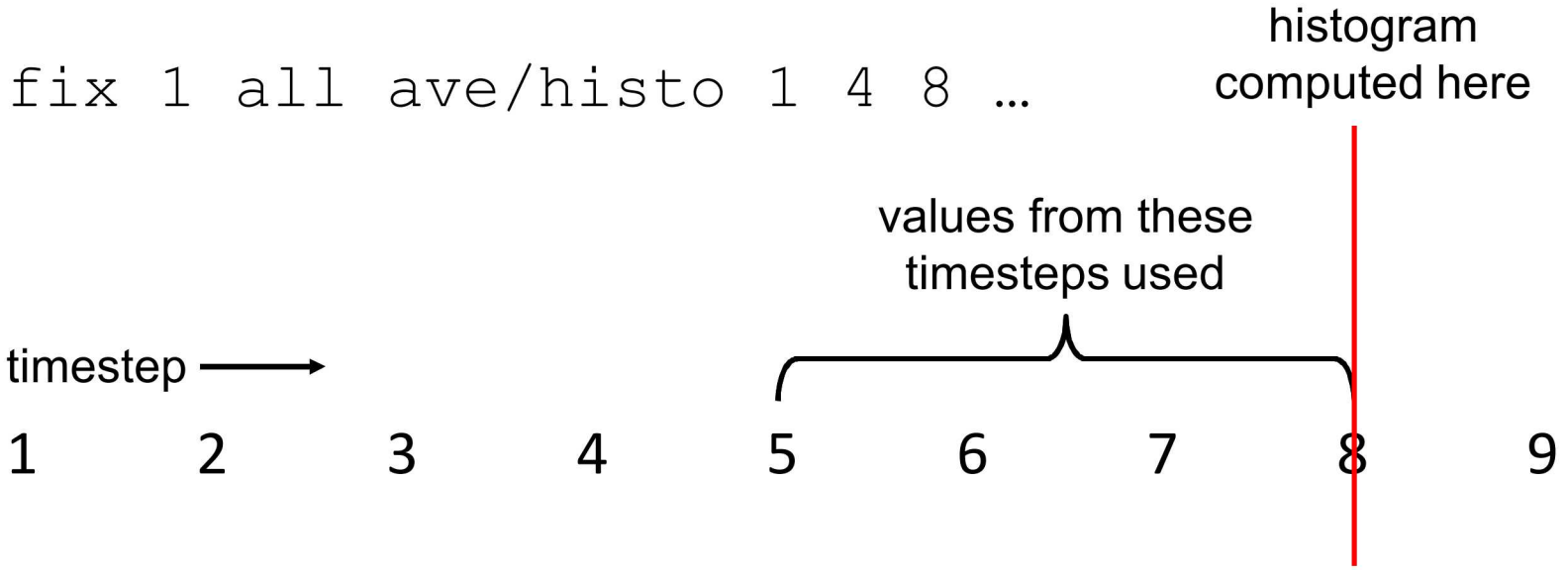
- Can use common math operators such as + and -, as well as common match functions such as sin() and cos()
- Can reference one variable in another variable
- Equal-style variables are scalars
  - `variable beta equal temp/3.0`
  - `variable b1 equal a[234]+0.5*vol`
  - `variable b2 equal sin(a)/2.0`
  - `variable b3 equal c_myTemp`
- Atom-style variables are arrays
  - `variable  atom x*y/vol`
- Other types of variables, see LAMMPS documentation

Averaging Frequency

```
fix 1 all ave/histo [nevery] [nrepeat] [nfreq]…
```

- Nevery = use input values every this many timesteps
- Nrepeat = # of times to use input values for calculating histogram
- Nfreq = calculate histogram every this many timesteps

```
fix 1 all ave/histo 1 4 8 …
```



LAMMPS Log File

Sandia National Laboratories

- LAMMPS log file contains thermodynamic information about the system (energy, pressure, volume, etc.)
- Can define custom variables and output those to log file
- Per-atom quantities are output in the “dump” file, not log file
- Log file also contains other information such as timing breakdown and estimated memory usage

Data Files

Sandia National Laboratories

- The data file contains basic information such as:
  - size of the problem to be run
  - the initial atomic coordinates
  - molecular topology
  - and (optionally) force-field coefficients
- Data files are text files
- Restart files are binary files which save a system configuration
- Can convert a restart file to a data file using the “-restart” command line option

Data file and molecule file formats

```
LAMMPS Description              (1st line of file)

100 atoms          (this must be the 3rd line, 1st 2 lines are ignored)
95 bonds                (# of bonds to be simulated)
50 angles               (include these lines even if number = 0)
30 dihedrals
20 impropers

5 atom types           (# of nonbond atom types)
10 bond types          (# of bond types = sets of bond coefficients)
18 angle types
20 dihedral types      (do not include a bond,angle,dihedral,improper type
2 improper types             line if number of bonds,angles,etc is 0)

-0.5 0.5 xlo xhi       (for periodic systems this is box size,
-0.5 0.5 ylo yhi        for non-periodic it is min/max extent of atoms)
-0.5 0.5 zlo zhi       (do not include this line for 2-d simulations)
```

Data file and molecule file formats

```
Masses

  1 mass
  ...
  N mass                              (N = # of atom types)

Nonbond Coeffs

  1 coeff1 coeff2 ...
  ...
  N coeff1 coeff2 ...                 (N = # of atom types)

Bond Coeffs

  1 coeff1 coeff2 ...
  ...
  N coeff1 coeff2 ...                 (N = # of bond types)
(N = # of angle types)
```

Data file and molecule file formats

```
Atoms

  1 molecule-tag atom-type q x y z nx ny nz   (nx,ny,nz are
optional -
  ...                                         see "true flag" input
command)
  ...
  N molecule-tag atom-type q x y z nx ny nz   (N = # of atoms)

Velocities

  1 vx vy vz
  ...
  ...
  N vx vy vz                                  (N = # of atoms)
```

Data file and molecule file formats

```
Bonds

  1 bond-type atom-1 atom-2
  ...
  N bond-type atom-1 atom-2          (N = # of bonds)

Angles

  1 angle-type atom-1 atom-2 atom-3  (atom-2 is the center atom in angle)
  ...
  N angle-type atom-1 atom-2 atom-3  (N = # of angles)

Dihedrals

  1 dihedral-type atom-1 atom-2 atom-3 atom-4  (atoms 2-3 form central bond)
  ...
  N dihedral-type atom-1 atom-2 atom-3 atom-4  (N = # of dihedrals)

Impropers

  1 improper-type atom-1 atom-2 atom-3 atom-4  (atom-2 is central atom)
  ...
  N improper-type atom-1 atom-2 atom-3 atom-4  (N = # of impropers)
```

Lattice Command

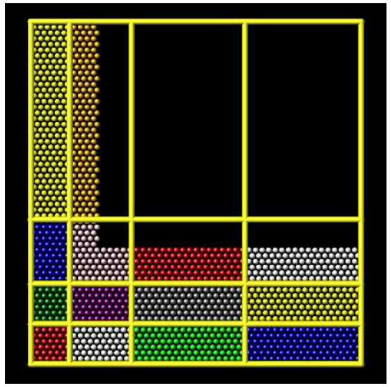
- In LAMMPS, a lattice is simply a set of points in space, determined by a unit cell with basis atoms, that is replicated infinitely in all dimensions
- "Create atoms" command creates atoms on the lattice points inside the simulation box
- For LJ units, the lattice command takes a reduced density, for other units, the lattice command takes the lattice constant

```
lattice       fcc 0.8442
region      box block 0 ${xx} 0 ${yy} 0 ${zz}
create_box    1 box
create_atoms  1 box
```

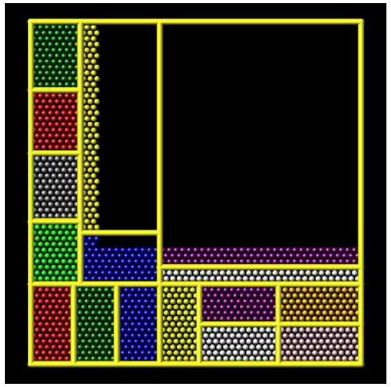
Load Balancing

- Adjusts the size and shape of processor sub-domains within the simulation box
- Attempts to balance the number of atoms or particles and thus indirectly the computational cost (load) more evenly across processors
- Can be static or dynamic

shift



rcb



```
balance 1.0 shift xz 5 1.1
balance 1.1 rcb
```

Accelerator Packages

Sandia National Laboratories

- Some hardware components like GPUs, Xeon Phis, and multithreaded CPUs require special code (i.e. OpenMP, CUDA) to fully take advantage of the hardware
- LAMMPS has 5 accelerator packages:
  - USER-OMP
  - USER-INTEL
  - GPU
  - KOKKOS
  - OPT
- See http://lammps.sandia.gov/doc/Section_accelerate.html (How to build and run with accelerator packages is beyond the scope of this tutorial)

Downloading LAMMPS

Sandia National Laboratories

- LAMMPS Website (http://lammps.sandia.gov)
  - Go to “download” link
  - Download gzipped tar file
- Github (https://github.com/lammps/lammps)
  - https://github.com/lammps/lammps/releases
  - Clone or download button, then download zip file
  - git clone … (beyond this tutorial)
- Stable version: more testing
- Development version: latest features and bug fixes

Compiling LAMMPS

Sandia National Laboratories

- Need C++ compiler (GNU, Intel, Clang)
- Need MPI library, or can use the “STUBS” library
- Many Makefiles in src/MAKE

```
$ cd src
$ make mpi # uses src/MAKE/Makefile.serial
```

- produces executable named “lmp_mpi” in the src directory
- LAMMPS also has an alternative CMake interface

Using the MPI STUBS Library

Sandia National Laboratories

- Allows one to build LAMMPS without an MPI library installed

```
$ cd src/STUBS
$ make
$ cd ..
$ make serial # uses src/MAKE/Makefile.serial
```

- produces executable named “lmp_serial” in the src directory

Optional Packages

Sandia National Laboratories

- LAMMPS is very modular and has several optional packages
- Install optional package: `make yes-[PACKAGE]`, i.e. `make yes-manybody`
- Uninstall optional package: `make no-[PACKAGE]`, i.e. `make no-manybody`
- See the LAMMPS documentation for whether or not you need an optional package
- Use the `-h` command-line option to see which packages are installed in a LAMMPS executable

Running LAMMPS

Sandia National Laboratories

- Syntax: `[executable] -in [input_script]`
- In serial:

```
$ ./lmp_serial -in in.lj
```

- In parallel:

```
$ mpirun -np 2 lmp_mpi -in in.lj
```

- Many other command line options, see http://lammps.sandia.gov/doc/Section_start.html#start-6

Common Errors

Sandia National Laboratories

Lost atoms or out of range atoms - cannot compute PPPM

- Possible causes are
  - bad initial geometry (close contacts), minimize first
  - bad force field parameters or a typo (atoms get too close and then one "shoots" through the system)
  - use of shrinkwrap boundary with lots of extra space (atoms get lost when subdomains move too much (happens only in parallel runs)
  - fixed boundaries without a wall
- To hide the issue, can use `thermo_modify lost ignore`

Common Errors (Continued)

NaN (not a number) in output

- For positions with variable cell: too high potential energy → start with fixed volume, run minimization first
- For positions with fixed volume: close contacts

Unknown XXX style = missing package

Illegal XXX style command = syntax error, often due to version mismatch of online vs. version specific docs

Energy/stress not tallied on current time step

- pe, pressure, pe/atom and stress/atom computes need to be “triggered" by a fix, dump or thermo output either directly or indirectly through a variable

LAMMPS Directories

Sandia National Laboratories

- The "examples" directory has many example inputs, good place to learn commands
- The "potentials" directory has different force field files, such as EAM, Tersoff, etc.
- The "tools" directory has tools, i.e. one can convert inputs from other codes such as CHARMM to LAMMPS
- The "lib" directory has optional libraries bundled with LAMMPS (may require compiling)
- The "bench" directory has canonical benchmarks used to test LAMMPS performance

Example: Lennard Jones

Sandia National Laboratories

1. Run default input script: bench/in.lj
2. Change initial velocity distribution to give higher temperature
3. Change integrator from constant energy (fix nve) to constant temperature (fix nvt)
4. Decrease density of the system
5. Dump atom positions and forces
6. Write out data file and restart file
7. Make a histogram of per-atom kinetic energy

Example: Lennard Jones

Sandia National Laboratories

Run default input script: bench/in.lj

```
$ cd bench
$ ../src/lmp_serial -in in.lj
```

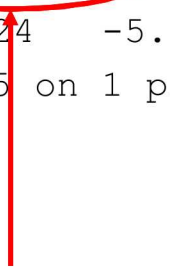
```
Step Temp E_pair E_mol TotEng Press
       0          1.44   -6.7733681             0   -4.6134356   -5.0197073
     100    0.7574531   -5.7585055             0   -4.6223613    0.20726105
Loop time of 3.19363 on 1 procs for 100 steps with 32000 atoms
```

Example: Lennard Jones

Change velocity distribution to give higher temperature

```
#velocity all create 1.44 87287 loop geom
velocity  all create 2.0 87287 loop geom
```

```
Step Temp E_pair E_mol TotEng Press
       0            2   -6.7733681            0   -3.7734618     -4.54697
     100    1.0614424   -5.3736027            0   -3.7814889    2.3907499
Loop time of 3.35405 on 1 procs for 100 steps with 32000 atoms
```



now 2.0 instead of 1.44

Example: Lennard Jones

Change integrator from constant energy (fix nve) to constant temperature (fix nvt)

```
#fix 1 all nve
fix         1 all nvt 1.0 1.0 0.1
```

```
Step Temp E_pair E_mol TotEng Press
       0          1.44   -6.7733681            0   -4.6134356   -5.0197073
     100   0.92987881   -5.5836548            0   -4.1888802    1.2277583
Loop time of 3.23382 on 1 procs for 100 steps with 32000 atoms
```

close to 1.0

Example: Lennard Jones

Sandia National Laboratories

Decrease density of the system

```
#lattice           fcc 0.8442
lattice           fcc 0.1
```

```
Step Temp E_pair E_mol TotEng Press
       0          1.44      -0.1194            0    2.0405325     0.1202355
     100       1.65803  -0.39464997            0    2.0923173    0.18136994
Loop time of 0.598197 on 1 procs for 100 steps with 32000 atoms
```

Example: Lennard Jones

Sandia National Laboratories

Dump atom positions and forces

```
dump myCustom all custom 100 dump.custom id x y
   z fx fy fz
```

```
ITEM: TIMESTEP
100
ITEM: NUMBER OF ATOMS
32000
ITEM: BOX BOUNDS pp pp pp
0.0000000000000000e+00 3.3591923827650149e+01
0.0000000000000000e+00 3.3591923827650149e+01
0.0000000000000000e+00 3.3591923827650149e+01
ITEM: ATOMS id x y z fx fy fz
1 0.227853 0.074729 0.0873785 -20.1615 9.18853 -5.79564
2 0.919519 0.846223 0.101134 0.57499 14.6108 3.75088
3 0.99976 0.0650636 0.951449 -2.22408 8.98378 -9.31871
...
```

Example: Lennard Jones

Sandia National Laboratories

Write out data file and restart file

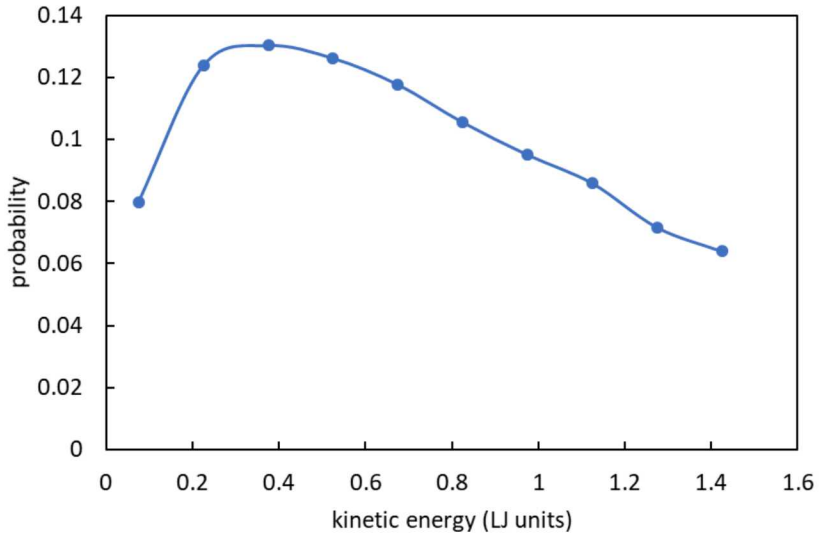
```
run                 100
write_data          out.data
write_restart       out.rst
```

```
LAMMPS data file via write_data, version 29 Jun 2018, timestep = 100

32000 atoms
1 atom types

0.0000000000000000e+00 3.3591923827650149e+01 xlo xhi
0.0000000000000000e+00 3.3591923827650149e+01 ylo yhi
0.0000000000000000e+00 3.3591923827650149e+01 zlo zhi
...
```

Example: Lennard Jones

Make a histogram of per-atom kinetic energy

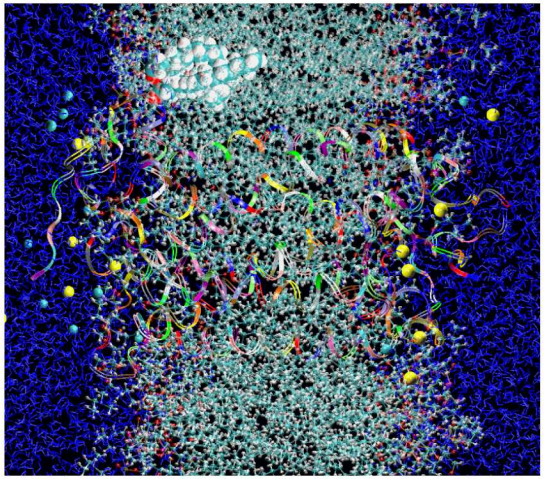
```
compute myKe all ke/atom
fix myHisto all ave/histo 1 1 100 0.0 1.5 10
  c_myKe file temp.histo mode vector
```



Example: Rhodopsin (Solvated Protein)

Sandia National Laboratories

1. Run default input script: bench/in.rhodo
2. Output thermodynamic quantities every timestep
3. Change pair_style to use cutoff instead of long-range electrostatics
4. Turn off fix shake



Example: Rhodopsin

Sandia National Laboratories

Compile LAMMPS with necessary packages

```
cd src
make yes-molecule # bonds, angles, etc.
make yes-kokkos # long-range electrostatics
make yes-rigid # fix shake
make mpi
```

Example: Rhodopsin

Sandia National Laboratories

Run default input script: bench/in.rhodo

```
cd bench
../src/lmp_mpi -in in.rhodo
```

```
---------------- Step      100 ----- CPU =     46.4527 (sec) ----------------
TotEng   =    -25290.7388 KinEng   =     21591.9096 Temp     =       301.0906
PotEng   =    -46882.6484 E_bond   =      2567.9789 E_angle  =     10781.9556
E_dihed  =      5198.7493 E_impro  =       216.7863 E_vdwl   =     -1902.6458
E_coul   =    206659.5006 E_long   =   -270404.9733 Press    =         6.7898
Volume   =    308133.9933
Loop time of 46.4528 on 1 procs for 100 steps with 32000 atoms
```

Example: Rhodopsin

Sandia National Laboratories

Output thermodynamic quantities every timestep

```
#thermo          50
thermo           1
```

```
---------------- Step         1 ----- CPU =      0.5639 (sec) ----------------
TotEng   =    -25351.0706 KinEng   =     21397.7565 Temp     =       298.3832
PotEng   =    -46748.8271 E_bond   =      2564.4411 E_angle  =     10943.8892
E_dihed  =      5216.2164 E_impro  =       207.5740 E_vdwl   =     -2305.6968
E_coul   =    207029.1842 E_long   =   -270404.4353 Press    =      -102.3522
Volume   =    307994.9726
```

Example: Rhodopsin

Change pair_style to use cutoff instead of long-range electrostatics (good idea?)

```
#pair_style        lj/charmm/coul/long 8.0 10.0
#kspace_style      pppm 1e-4
pair_style        lj/charmm/coul/charmm 8.0 10.0
```

```
---------------- Step      100 ----- CPU =     43.4577 (sec) ----------------
TotEng   =    -24121.6022 KinEng   =     21717.0422 Temp     =       302.8355
PotEng   =    -45838.6444 E_bond   =      2495.7190 E_angle  =     10583.2010
E_dihed  =      5182.0276 E_impro  =       213.9166 E_vdwl   =     -2044.3302
E_coul   =    -62269.1784 E_long   =         0.0000 Press    =       -64.9369
Volume   =    308128.5584
```

Example: Rhodopsin

Sandia National Laboratories

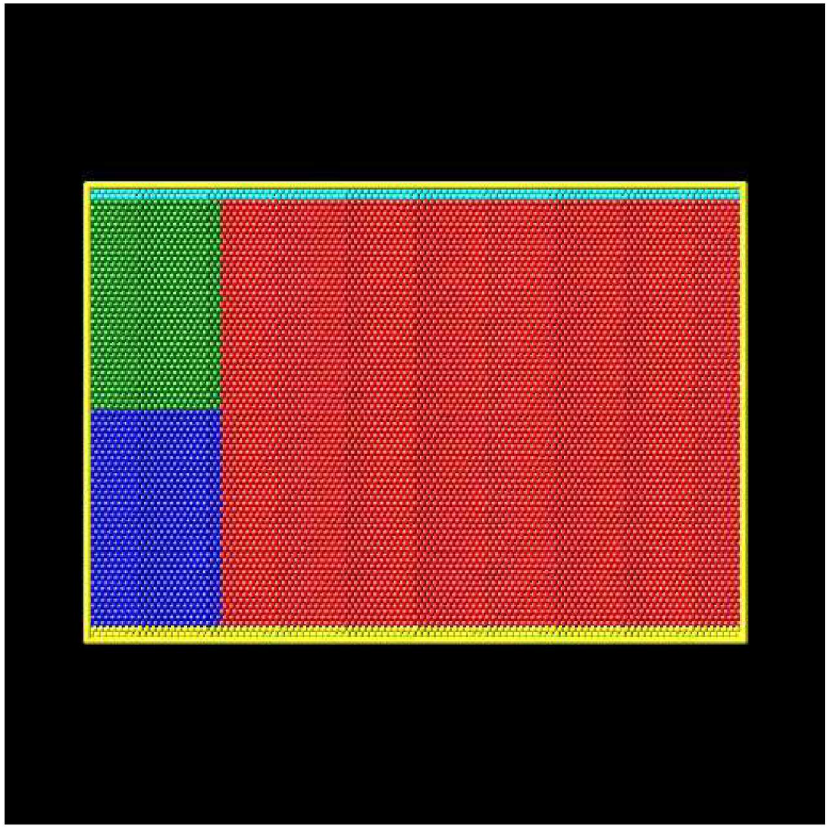
Turn off fix shake (good idea?)

```
#fix             1 all shake 0.0001 5 0 m 1.0 a 232
```

```
---------------- Step      100 ----- CPU =     46.5689 (sec) ----------------
TotEng   =     -8947.2550 KinEng   =     31456.5029 Temp     =       329.7917
PotEng   =    -40403.7580 E_bond   =      6018.3250 E_angle  =     15154.4978
E_dihed  =      5741.2343 E_impro  =       283.1834 E_vdwl   =      -176.9129
E_coul   =    202951.6708 E_long   =   -270375.7564 Press    =      -166.9381
Volume   =    307809.0050
```

Example: Crack

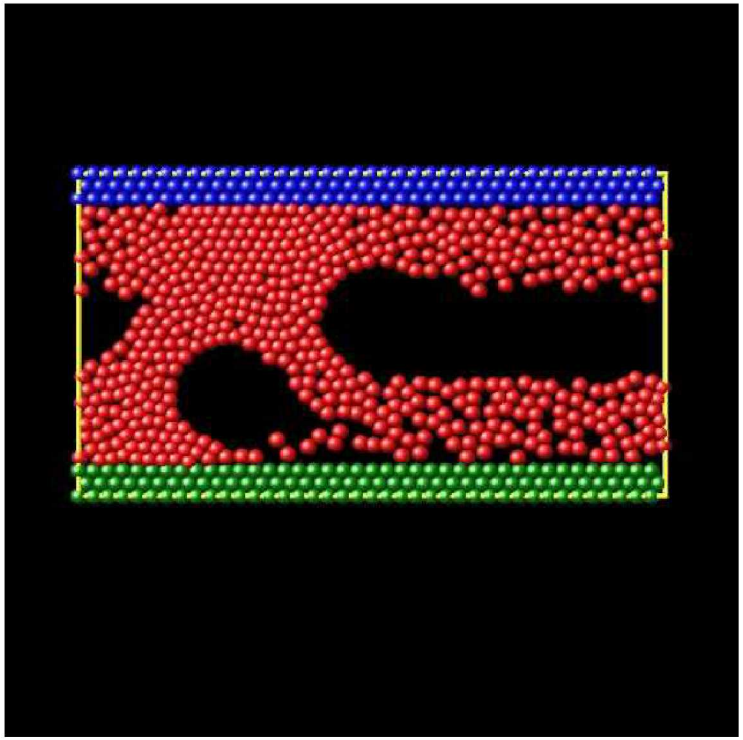
Run the crack example interactively



Example: Obstacle

Sandia National Laboratories

Run the obstacle example interactively



Example of Extending LAMMPS

Sandia National Laboratories

- I wanted to use fix efield and an Ewald sum with point dipoles but it wasn't yet supported
- I emailed the LAMMPS mail list to ask if anyone was working on an Ewald sum (to not duplicate work)
- I consulted with Steve Plimpton (LAMMPS developer) for advice
- I used the existing code as a starting point/pattern and then added my own code, following the existing style
- I tested the code and updated doc page and when done sent it to Steve to be included in LAMMPS so that others could use it too

Use Existing Code First

Sandia National Laboratories

- Steve sent me some files for the dipole/long pair_style that included support for the long-range Ewald sum (some issues needed to be fixed)
- There was already some code for this already in the kspace_style ewald/disp (some issues needed to be fixed)
- I used the existing fix efield and added support for dipoles there (didn't need to make a new fix)

Example of Extending LAMMPS

Sandia National Laboratories

#	Line 111	Line 131
131		
132	void FixEfield::init()	void FixEfield::init()
133	{	{
134	if (!atom->q_flag) error->all(FLERR,"Fix efield requires atom attribute q");	qflag = muflag = 0;
135		if (atom->q_flag) qflag = 1;
136		if (atom->mu_flag && atom->torque_flag) muflag = 1;
137		if (!qflag && !muflag)
138		error->all(FLERR,"Fix efield requires atom attribute q or mu");
139		
140	// check variables	// check variables
141		
#	**Line 139**	**Line 163**
163	else if (input->variable->atomstyle(zvar)) zstyle = ATOM;	else if (input->variable->atomstyle(zvar)) zstyle = ATOM;
164	else error->all(FLERR,"Variable for fix efield is invalid style");	else error->all(FLERR,"Variable for fix efield is invalid style");
165	}	}
166		if (estr) {
167		evar = input->variable->find(estr);
168		if (evar < 0)
169		error->all(FLERR,"Variable name for fix efield does not exist");
170		if (input->variable->atomstyle(evar)) estyle = ATOM;
171		else error->all(FLERR,"Variable for fix efield is invalid style");
172		} else estyle = NONE;
173		
174	if (xstyle == ATOM || ystyle == ATOM || zstyle == ATOM)	if (xstyle == ATOM || ystyle == ATOM || zstyle == ATOM)
175	varflag = ATOM;	varflag = ATOM;
#	**Line 146**	**Line 177**
177	varflag = EQUAL;	varflag = EQUAL;
178	else varflag = CONSTANT;	else varflag = CONSTANT;
179		
180		if (muflag && varflag == ATOM)
181		error->all(FLERR,"Fix efield with dipoles cannot use atom-style variables");
182		
183		if (varflag == CONSTANT && estyle != NONE)
184		error->all(FLERR,"Cannot use variable energy with "
185		"constant efield in fix efield");
186		if ((varflag == EQUAL || varflag == ATOM) &&
187		update->whichflag == 2 && estyle == NONE)
188		error->all(FLERR,"Must use variable energy with fix efield");
189		

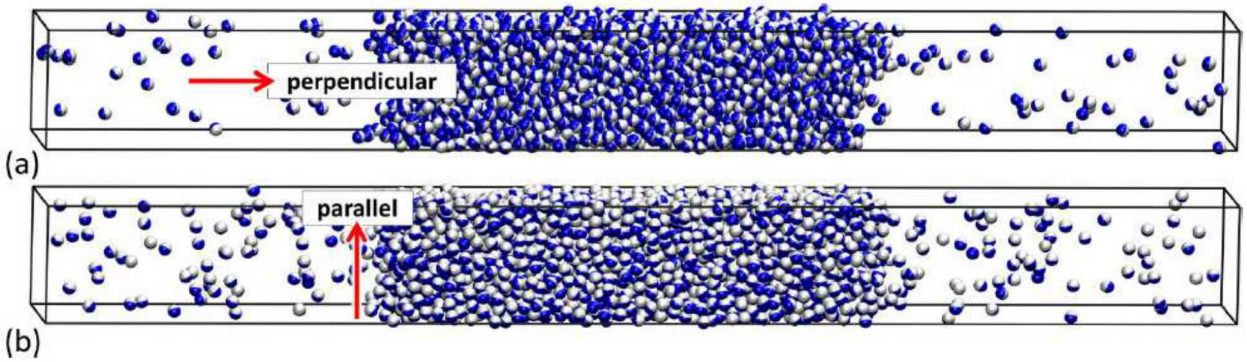
Example of Extending LAMMPS

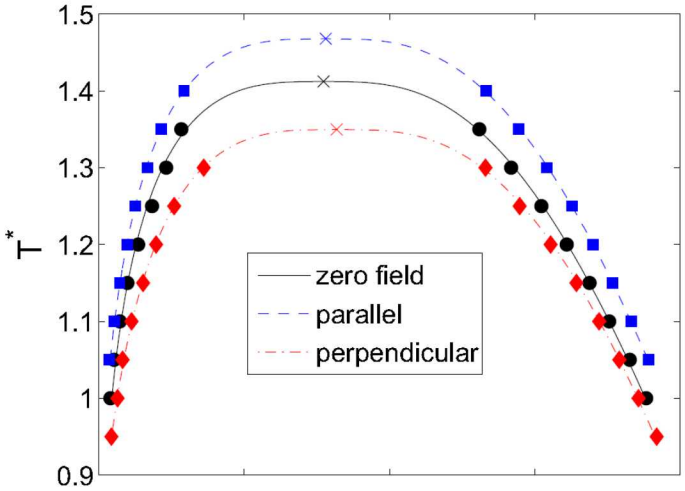
```
243                                                  // constant efield
244
245  if (varflag == CONSTANT) {                      if (varflag == CONSTANT) {
246    for (int i = 0; i < nlocal; i++)                double unwrap[3];
247      if (mask[i] & groupbit) {
248        fx = q[i]*ex;                               // charge interactions
249        fy = q[i]*ey;                               // force = qE, potential energy = F dot x in unwrapped coords
250        fz = q[i]*ez;
251        f[i][0] += fx;                              if (qflag) {
252        f[i][1] += fy;                                for (int i = 0; i < nlocal; i++)
253        f[i][2] += fz;                                  if (mask[i] & groupbit) {
254                                                          fx = q[i]*ex;
255        fsum[0] -= fx*x[i][0]+fy*x[i][1]+fz*x[i][2];      fy = q[i]*ey;
256        fsum[1] += fx;                                    fz = q[i]*ez;
257        fsum[2] += fy;                                    f[i][0] += fx;
258        fsum[3] += fz;                                    f[i][1] += fy;
259      }                                                   f[i][2] += fz;
260
261                                                          domain->unmap(x[i],image[i],unwrap);
262                                                          fsum[0] -= fx*unwrap[0]+fy*unwrap[1]+fz*unwrap[2];
263                                                          fsum[1] += fx;
264                                                          fsum[2] += fy;
265                                                          fsum[3] += fz;
266                                                        }
267                                                      }
268
269                                                    // dipole interactions
270                                                    // no force, torque = mu cross E, potential energy = -mu dot E
271
272                                                    if (muflag) {
273                                                      double **mu = atom->mu;
274                                                      double **t = atom->torque;
275                                                      double tx,ty,tz;
276                                                      for (int i = 0; i < nlocal; i++)
277                                                        if (mask[i] & groupbit) {
278                                                          tx = ez*mu[i][1] - ey*mu[i][2];
279                                                          ty = ex*mu[i][2] - ez*mu[i][0];
280                                                          tz = ey*mu[i][0] - ex*mu[i][1];
281                                                          t[i][0] += tx;
282                                                          t[i][1] += ty;
283                                                          t[i][2] += tz;
284                                                          fsum[0] -= mu[i][0]*ex + mu[i][1]*ey + mu[i][2]*ez;
285                                                        }
286                                                    }
```

$$\mathbf{T}_i^{\text{ext}} = \boldsymbol{\mu}_i \times \mathbf{E}^{\text{ext}}$$

Example of Extending LAMMPS: Results

- The field shifts the critical temperature of a Stockmayer fluid up or down, depending on the direction of the field relative to the interface





Liquid-vapor interface of the Stockmayer fluid in a uniform external field, Moore, Stevens, and Grest, Phys. Rev. E **91**, 022309 (2015)

Sandia National Laboratories

Questions?