A Quick Tour of LAMMPS

Steve Plimpton
Sandia National Labs
sjplimp@sandia.gov

6th LAMMPS Workshop Beginners Tutorial
August 2019 - Albuquerque, NM

Follow along with my slides at:
http://lammps.sandia.gov/workshops/Aug19/workshop.html

Sandia National Laboratories is a multi-mission 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-NA0003525. Presentation: SAND2017-7985C

Resources for learning LAMMPS

All on web site, many in distro tarball or GHub download:

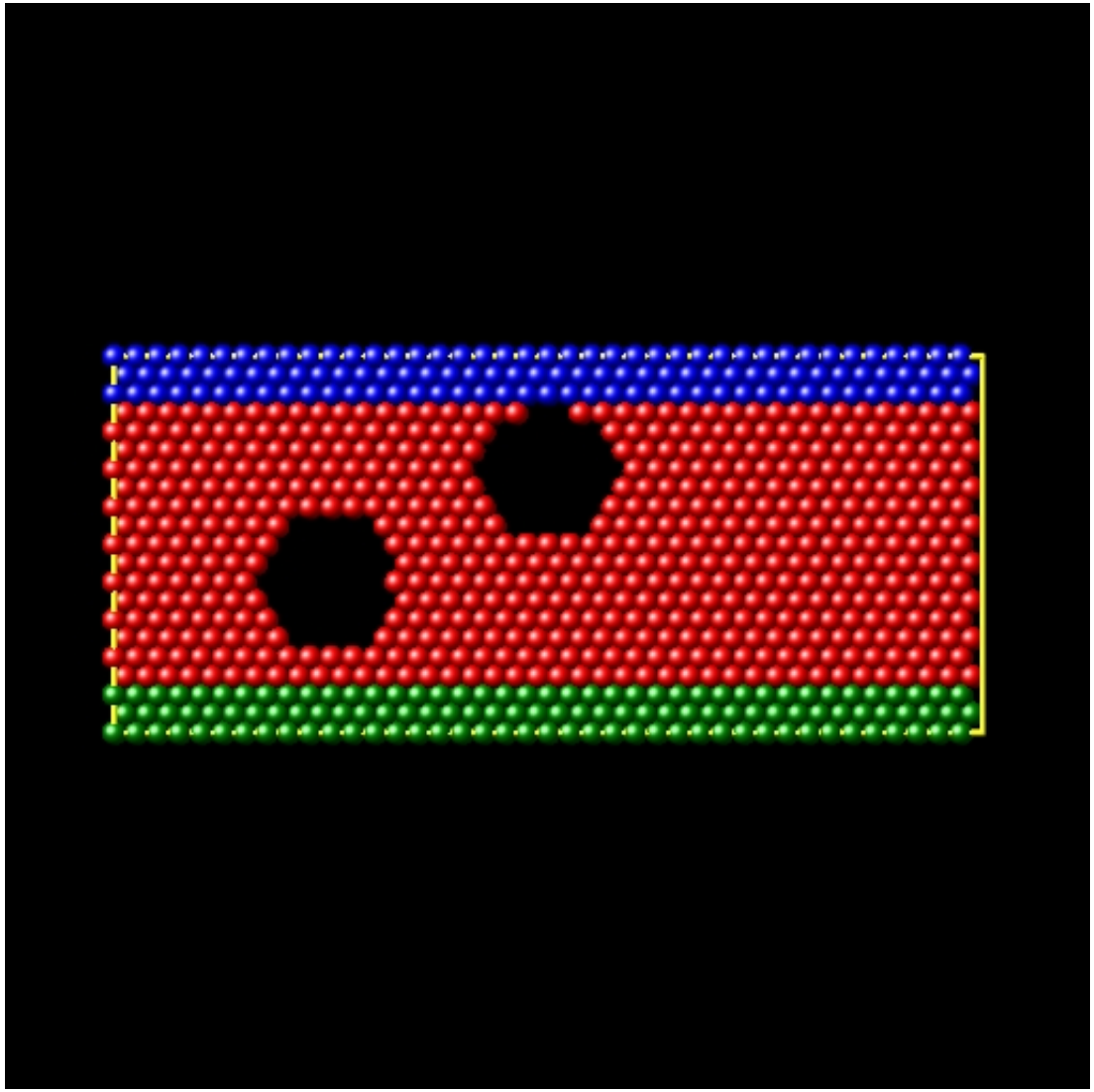
- Workshops (beginner tutorials) & Tutorials
- Features: list of features with links
- Glossary: MD terms ⇒ LAMMPS
- Manual:
  - Intro, Commands, Packages, Accelerating, Howto, Modifying, Errors pages
  - search bubble at top left of every page
- Commands (bookmark it!)
  - alphabetized command & style tables: one page per command
- Papers: find a paper similar to what you want to model
- Mail list: post Qs to it, subscribe
  - Google search: include lammps-users in search
  - http://lammps.sandia.gov/mail.html
- Examples: 600 input scripts under examples in distro
  - lower-case dirs = simple, upper-case = more complex
  - many produce movies: http://lammps.sandia.gov/movies.html

Structure of a typical input script

1. Units, dimension, atom style (optional)
2. Create simulation box and atoms
   - region, create_box, create_atoms, region commands
     - lattice command vs box units
   - read_data command
     - data file is a text file
     - look at examples/micelle/data.micelle
     - see read_data doc page for full syntax
3. Define groups (optional)
4. Set atom attributes: velocity, mass (may be in data file)
5. Pair style for atom interactions
6. Fixes for time integration and constraints
7. Computes for diagnostics
8. Output: thermo, dump, restart
9. Run or minimize
10. Rinse and repeat (script executed one command at a time)

Obstacle example

input script = examples/obstacle/in.obstacle



Obstacle input script

```
# 2d LJ obstacle flow (section 1)

dimension 2
boundary p s p
atom_style atomic
neighbor 0.3 bin
neigh_modify delay 5

# create box and atoms (section 2)

lattice hex 0.7
region box block 0 40 0 10 -0.25 0.25
create_box 3 box # 3 atom types
create_atoms 1 box
```

Obstacle input script (2)

```
# define groups and types (section 3)

region 1 block INF INF INF 1.25 INF INF
group lower region 1
region 2 block INF INF 8.75 INF INF INF
group upper region 2
group boundary union lower upper
group flow subtract all boundary

set group lower type 2
set group upper type 3

# LJ potential (section 5)
pair_style lj/cut 1.12246
pair_coeff * * 1.0 1.0 1.12246
```

Obstacle input script (3)

```
# initial velocities (section 4)

mass * 1.0
compute mobile flow temp
velocity flow create 1.0 482748 temp mobile
fix 1 all nve
fix 2 flow temp/rescale 200 1.0 1.0 0.02 1.0
fix_modify 2 temp mobile

# Poiseuille flow (section 6)

velocity boundary set 0.0 0.0 0.0
fix 3 lower setforce 0.0 0.0 0.0
fix 4 upper setforce 0.0 NULL 0.0
fix 5 upper aveforce 0.0 -0.5 0.0
fix 6 flow addforce 1.0 0.0 0.0
```

Obstacle input script (4)

```
# create 2 obstacles to flow (section 6)

region void1 sphere 10 4 0 3
delete_atoms region void1
region void2 sphere 20 7 0 3
delete_atoms region void2

fix 7 flow indent 100 sphere 10 4 0 4
fix 8 flow indent 100 sphere 20 7 0 4
fix 9 all enforce2d
```

Obstacle input script (5)

```
# define output and run (sections 8,9)

timestep 0.003
thermo 1000
thermo_modify temp mobile

#dump 1 all atom 100 dump.obstacle
dump 2 all image 500 image.*.ppm type type &
    zoom 1.6 adiam 1.5
dump_modify 2 pad 5

run 25000
```

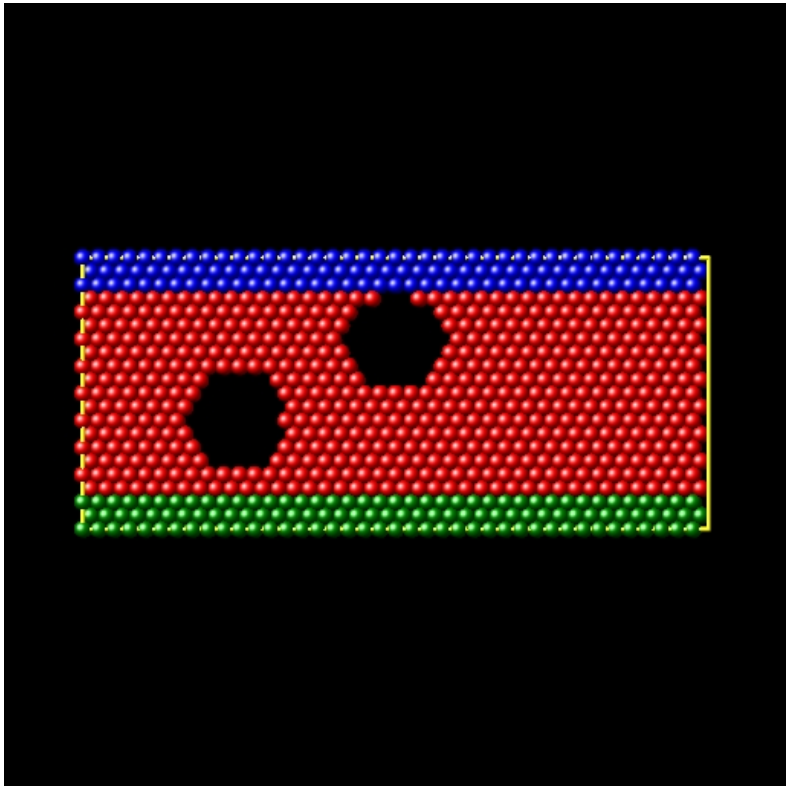
Make a movie

50 JPG or PNG or PPM files

- image.16500.ppm
- ImageMagick display
- Mac Preview

Make/view a movie

- ImageMagick
  convert *.ppm image.gif
- open in browser
  open -a Safari image.gif
- Mac QuickTime
  open image sequence
- Windows Media Player
- Ovito (redThurs: 1:15),
  VMD, AtomEye, ...



Obstacle example

Questions on input script?

Examine screen output

```
LAMMPS (7 Aug 2019)
Lattice spacing in x,y,z = 1.28436 2.22457 1.28436
Created orthogonal box = (0 0 -0.321089) to (51.3743
22.2457 0.321089)
  4 by 1 by 1 MPI processor grid
Created 840 atoms
  create_atoms CPU = 0.000680923 secs
120 atoms in group lower
120 atoms in group upper
240 atoms in group boundary
600 atoms in group flow
Setting atom values ...
  120 settings made for type
Setting atom values ...
  120 settings made for type
Deleted 36 atoms, new total = 804
Deleted 35 atoms, new total = 769
```

Neighbor list and memory info

```
Neighbor list info ...
  update every 1 steps, delay 5 steps, check yes
  max neighbors/atom:  2000, page size:  100000
  master list distance cutoff = 1.42246
  ghost atom cutoff = 1.42246
  binsize = 0.71123, bins = 73 32 1
  1 neighbor lists, perpetual/occasional/extra = 1 0 0
  (1) pair lj/cut, perpetual
    attributes:  half, newton on
    pair build:  half/bin/atomonly/newton
    stencil:  half/bin/2d/newton
    bin:  standard
Setting up Verlet run ...
  Unit style :  lj
  Current step :  0
  Time step :  0.003
Per MPI rank memory allocation (min/avg/max) =
  3.043 | 3.044 | 3.044 Mbytes
```

Thermo output

```
Step Temp E_pair E_mol TotEng Press Volume
0 1.0004177 0 0 0.68689281 0.46210058 1143.0857
1000 1 -0.32494012 0 0.36166587 1.2240503 1282.5239
2000 1 -0.37815616 0 0.30844982 1.0642877 1312.5691
...
...
...
24000 1 -0.40040333 0 0.28620265 0.94983886 1459.4461
25000 1 -0.37645924 0 0.31014674 1.0526044 1458.7191
Loop time of 0.81263 on 4 procs for 25000 steps with
769 atoms
Performance:  7974111.120 tau/day,
  30764.318 timesteps/s
98.0% CPU use with 4 MPI tasks x no OpenMP threads
```

Timing info

```
Loop time of 0.815388 on 4 procs for 25000 steps
  with 769 atoms

MPI task timing breakdown:
Section | min | avg | max | %varavg | %total
---------------------------------------------
Pair | 0.063 | 0.123 | 0.202 | 15.7 | 15.10
Neigh | 0.031 | 0.041 | 0.055 | 4.4 | 5.06
Comm | 0.149 | 0.231 | 0.288 | 11.0 | 28.42
Output | 0.001 | 0.001 | 0.001 | 0.0 | 0.01
Modify | 0.275 | 0.305 | 0.341 | 4.3 | 37.43
Other | ----- | 0.113 | ----- | --- | 13.91
```

Run statistics

Per-processor values at end of run

```
Nlocal:  192.25 ave 243 max 151 min
Histogram:  1 1 0 0 0 0 1 0 0 1
Nghost:  41.75 ave 43 max 39 min
Histogram:  1 0 0 0 0 0 0 1 0 2
Neighs:  408.5 ave 575 max 266 min
Histogram:  1 1 0 0 0 0 0 1 0 1

Total # of neighbors = 1634
Ave neighs/atom = 2.12484
Neighbor list builds = 1631
Dangerous builds = 1
```

Run statistics

Per-processor values at end of run

```
Nlocal:  192.25 ave 243 max 151 min
Histogram:  1 1 0 0 0 0 1 0 0 1
Nghost:  41.75 ave 43 max 39 min
Histogram:  1 0 0 0 0 0 0 1 0 2
Neighs:  408.5 ave 575 max 266 min
Histogram:  1 1 0 0 0 0 0 1 0 1

Total # of neighbors = 1634
Ave neighs/atom = 2.12484
Neighbor list builds = 1631
Dangerous builds = 1
```

Questions on output?

Common input script errors and debugging

- Command not recognized
  - ERROR: Unrecognized pair style 'gran/hooke/history' is part of the GRANULAR package which is not enabled in this LAMMPS binary
  - LAMMPS was not built with package for style being used
  - lmp_mpi -h will list all packages & styles included in build
- Syntax error in command
  - ERROR: Incorrect args for pair coeffs (pair_lj_cut.cpp:447)
  - Last command: pair_coeff * * 1.0 1.0 1.12246 9.0

Common input script errors and debugging

- Command not recognized
  - ERROR: Unrecognized pair style 'gran/hooke/history' is part of the GRANULAR package which is not enabled in this LAMMPS binary
  - LAMMPS was not built with package for style being used
  - lmp_mpi -h will list all packages & styles included in build
- Syntax error in command
  - ERROR: Incorrect args for pair coeffs (pair_lj_cut.cpp:447)
  - Last command: pair_coeff * * 1.0 1.0 1.12246 9.0
- Thermo output blows up immediately
  - usually due to bad atom coords and/or bad force field
  - often leads to lost or out-of-range atoms
- Don't start with a complex script
  - debug an input script like a computer program
  - start simple, add complexity one command at a time
  - use variables and print command to examine quantities
  - use thermo/dump output and viz to verify correctness

Remaining topics

1. More complex input scripts
2. Pre-processing tools to build a complex system
3. Force fields: pair, bond, and kspace styles
4. Fixes and computes
5. Output
6. Parallelization and performance
7. Quick tour of more advanced topics
8. Customizing and modifying LAMMPS

Defining variables in input scripts

- Styles: index, loop, equal, atom, python, file, (14 types)
  - variable x index run1 run2 run3 run4
  - variable x loop 100
  - variable x equal trap(f_JJ[3])*${scale}
  - variable x atom -(c_p[1]+c_p[2]+c_p[3])/(3*vol)

Defining variables in input scripts

- Styles: index, loop, equal, atom, python, file, (14 types)
  - variable x index run1 run2 run3 run4
  - variable x loop 100
  - variable x equal trap(f_JJ[3])*${scale}
  - variable x atom -(c_p[1]+c_p[2]+c_p[3])/(3*vol)
- Formulas can be complex
  - see doc/variable.html
  - thermo keywords (temp, press, ...)
  - math operators & functions (sqrt, log, cos, ...)
  - group and region functions (count, xcm, fcm, ...)
  - various special functions (min, ave, trap, stride, stagger, ...)
  - per-atom vectors (x, vx, fx, ...)
  - output from computes, fixes, other variables
- Formulas can be time- and/or spatially-dependent

Using variables in input scripts

- Substitute in any command via $x or ${myVar}
- Immediate formula evaluation via $() syntax:
  - avoids need to define variable separately
  - variable xmid equal (xlo+xhi)/2
  - region 1 block $xmid EDGE INF INF EDGE EDGE
  - region 1 block $((xlo+xhi)/2) EDGE INF INF EDGE EDGE
- Next command increments a variable to next value
- Many commands allow variables as arguments
  - only if command doc page says so
  - fix addforce 0.0 v_fy 1.0
  - dump_modify every v_foo
  - region sphere 0.0 0.0 0.0 v_radius

Power tools for input scripts

- Filename options:
  - dump.*.% for per-snapshot or per-processor output
  - read_data data.protein.gz
  - read_restart old.restart.*
- If/then/else via if command
- Insert another script via include command
  - useful for long list of saved params (e.g. force field coeffs)

Power tools for input scripts

- Filename options:
  - dump.*.% for per-snapshot or per-processor output
  - read_data data.protein.gz
  - read_restart old.restart.*
- If/then/else via if command
- Insert another script via include command
  - useful for long list of saved params (e.g. force field coeffs)
- Looping via next and jump commands
- Invoke a shell command or external program
  - shell cd subdir1
  - shell my_analyze out.file $n ${param}
- Invoke Python from your script: doc/Python_head.html
  - pass LAMMPS data to Python, return values in variables
  - Python function can callback to LAMMPS
- Various ways to run multiple simulations from one script
  - see doc/Howto_multiple.html

Example script for multiple runs

Run 8 successive simulations on P processors:

```
variable r equal random(1,1000000000,58798)
variable a loop 8
variable t index 0.8 0.85 0.9 0.95 1.0 1.05 1.1 1.15
log log.$a
read data.polymer
velocity all create $t $r
fix 1 all nvt $t $t 1.0
dump 1 all atom 1000 dump.$a.*
run 100000
next t
next a
jump in.polymer
```

Example script for multiple runs

Run 8 successive simulations on P processors:

```
variable r equal random(1,1000000000,58798)
variable a loop 8
variable t index 0.8 0.85 0.9 0.95 1.0 1.05 1.1 1.15
log log.$a
read data.polymer
velocity all create $t $r
fix 1 all nvt $t $t 1.0
dump 1 all atom 1000 dump.$a.*
run 100000
next t
next a
jump in.polymer
```

Or instead, run 8 simulations on 3 partitions until finished:

- change a & t to universe-style variables
- mpirun -np 12 lmp_linux -p 3x4 -in in.polymer

Pre-processing tools to build complex systems

LAMMPS does not build molecular systems or auto-magically assign force field parameters for you

Pre-processing tools to build complex systems

LAMMPS does not build molecular systems or auto-magically assign force field parameters for you

- Data file must include list of bonds, angles, etc
- Data file can optionally include force field assignments
- Tools directory has converters for both steps
  - ch2lmp = CHARMM converter
  - amber2lmp = AMBER converter
  - msi2lmp = Accelrys converter

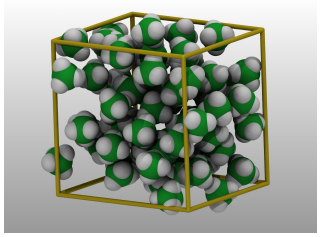
Pre-processing tools to build complex systems

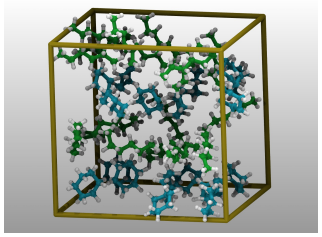
LAMMPS does not build molecular systems or auto-magically assign force field parameters for you

- Data file must include list of bonds, angles, etc
- Data file can optionally include force field assignments
- Tools directory has converters for both steps
  - ch2lmp = CHARMM converter
  - amber2lmp = AMBER converter
  - msi2lmp = Accelrys converter
- Builders that can create LAMMPS input
  - Pre/Post processing website page (free & commercial)
  - Pizza.py = chain and patch tools (Python)
  - VMD TopoTools (Axel Kohlmeyer, today: 2 PM)
  - Moltemplate (Andrew Jewett, Thurs: 1:15 PM)
  - Enhanced Monte Carlo (Pieter in 't Veld, Thurs: 4:30 PM)
  - Avogadro, Packmol, ATB (Auto Topology Builder)

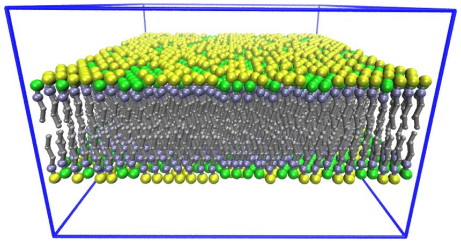
Builder tool examples

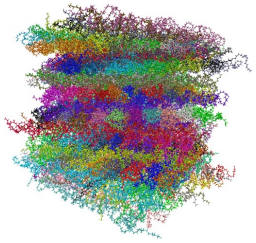
VMD TopoTools:

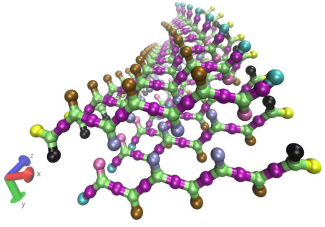




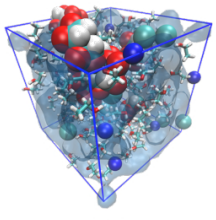
Moltemplate:



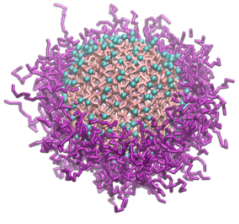




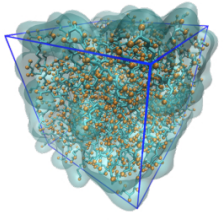
EMC:



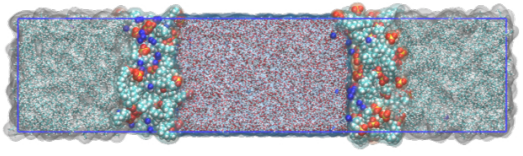
bulk



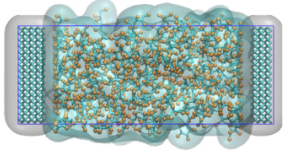
graft



copolymer



multiphase



surface

Pair styles

LAMMPS lingo for interaction potentials or force fields

Pair styles

LAMMPS lingo for interaction potentials or force fields

- A pair style can be true pair-wise or many-body
  - LJ, Coulombic, Buckingham, Morse, Yukawa, ...
  - EAM, Tersoff, REBO, ReaxFF, ...
- Bond/angle/dihedral/improper styles = permanent bonds

Pair styles

LAMMPS lingo for interaction potentials or force fields

- A pair style can be true pair-wise or many-body
  - LJ, Coulombic, Buckingham, Morse, Yukawa, ...
  - EAM, Tersoff, REBO, ReaxFF, ...
- Bond/angle/dihedral/improper styles = permanent bonds
- Variants optimized for CPU (many-core), KNL, GPU
  - OPT, GPU, USER-OMP, USER-INTEL, KOKKOS packages
  - lj/cut, lj/cut/gpu, lj/cut/kk, lj/cut/omp
  - see doc/Speed.html
- Coulomb interactions included in pair style
  - lj/cut, lj/cut/coul/cut, lj/cut/coul/wolf, lj/cut/coul/long
  - done to optimize inner loop

Many categories of pair styles

- Solids
  - eam, eim, meam, adp, etc
- Bio and polymers
  - charmm, class2, gromacs, dreiding, etc
- Reactive or bond-order
  - tersoff, bop, airebo, comb, reax/c, etc
- Coarse-grained
  - dpd, granular, sph, peri, colloid, lubricate, brownian, FLD
- Aspherical
  - gayberne, resquared, line, tri, etc

Many categories of pair styles

- Solids
  - eam, eim, meam, adp, etc
- Bio and polymers
  - charmm, class2, gromacs, dreiding, etc
- Reactive or bond-order
  - tersoff, bop, airebo, comb, reax/c, etc
- Coarse-grained
  - dpd, granular, sph, peri, colloid, lubricate, brownian, FLD
- Aspherical
  - gayberne, resquared, line, tri, etc

- Pair table for tabulation of any pair-wise interaction
- Pair hybrid style enables for hybrid models
  - polymers on metal
  - CNTs in water
  - solid-solid interface between 2 materials

Pair styles

See doc/Commands_pair.html for full list
Annotated with (gikot) for 5 accelerated variants

5.8. Pair_style potentials

All LAMMPS pair_style commands. Some styles have accelerated versions. This is indicated by additional letters in parenthesis: g = GPU, i = USER-INTEL, k = K

none	zero	hybrid (k)	hybrid/overlay (k)
adp (o)	agni (o)	airebo (io)	airebo/morse (io)
atm	awpmd/cut	beck (go)	body/nparticle
body/rounded/polygon	body/rounded/polyhedron	bop	born (go)
born/coul/dsf	born/coul/dsf/cs	born/coul/long (go)	born/coul/long/cs (g)
born/coul/msm (o)	born/coul/wolf (go)	born/coul/wolf/cs (g)	brownian (o)
brownian/poly (o)	buck (giko)	buck/coul/cut (giko)	buck/coul/long (giko)
buck/coul/long/cs	buck/coul/msm (o)	buck/long/coul/long (o)	buck/mdf
buck6d/coul/gauss/dsf	buck6d/coul/gauss/long	colloid (go)	comb (o)
comb3	coul/cut (gko)	coul/cut/soft (o)	coul/debye (gko)
coul/diel (o)	coul/dsf (gko)	coul/long (gko)	coul/long/cs (g)
coul/long/soft (o)	coul/msm (o)	coul/shield	coul/streitz
coul/wolf (ko)	coul/wolf/cs	dpd (gio)	dpd/fdt
dpd/fdt/energy (k)	dpd/tstat (go)	dsmc	e3b
drip	eam (gikot)	eam/alloy (gikot)	eam/cd (o)
eam/cd/old (o)	eam/fs (gikot)	edip (o)	edip/multi

Pair styles

See doc/pair_style.html for one-line descriptions

- none - turn off pairwise interactions
- hybrid - multiple styles of pairwise interactions
- hybrid/overlay - multiple styles of superposed pairwise interactions
- zero - neighbor list but no interactions
- adp - angular dependent potential (ADP) of Mishin
- agni - machine learned potential mapping atomic environment to forces
- airebo - AIREBO potential of Stuart
- airebo/morse - AIREBO with Morse instead of LJ
- atm - Axilrod-Teller-Muto potential
- awpmd/cut - Antisymmetrized Wave Packet MD potential for atoms and electrons
- beck - Beck potential
- body/nparticle - interactions between body particles
- body/rounded/polygon - granular-style 2d polygon potential
- body/rounded/polyhedron - granular-style 3d polyhedron potential
- bop - BOP potential of Pettifor
- born - Born-Mayer-Huggins potential
- born/coul/dsf - Born with damped-shifted-force model
- born/coul/dsf/cs - Born with damped-shifted-force and core/shell model
- born/coul/long - Born with long-range Coulombics
- born/coul/long/cs - Born with long-range Coulombics and core/shell
- born/coul/msm - Born with long-range MSM Coulombics

Relative CPU cost of potentials is dramatic

See http://lammps.sandia.gov/bench.html#potentials for details
Can estimate how long your simulation will run

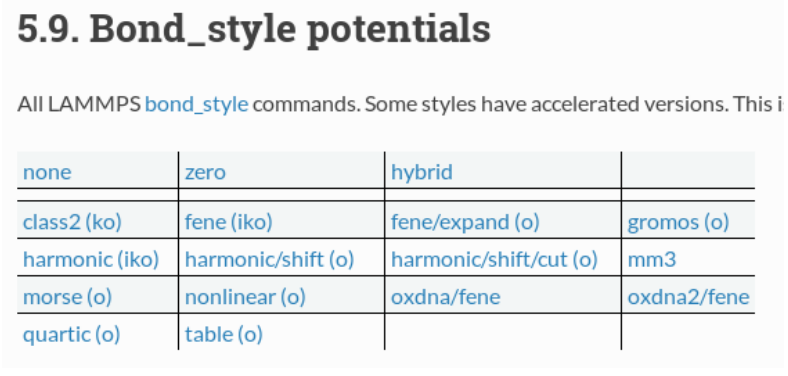
Potential	System	# Atoms	Timestep	Neighs/atom	Memory	CPU	LJ Ratio	P
Granular	chute flow	32000	0.0001 tau	7.2	33 Mb	2.08e-7	0.26x	
FENE bead/spring	polymer melt	32000	0.012 tau	9.7	8.4 Mb	2.86e-7	0.36x	
Lennard-Jones	LJ liquid	32000	0.005 tau	76.9	12 Mb	8.01e-7	1.0x	
DPD	pure solvent	32000	0.04 tau	41.3	9.4 Mb	1.22e-6	1.53x	
EAM	bulk Cu	32000	5 fmsec	75.5	13 Mb	1.87e-6	2.34x	
REBO	polyethylene	32640	0.5 fmsec	149	33 Mb	3.18e-6	3.97x	
Stillinger-Weber	bulk Si	32000	1 fmsec	30.0	11 Mb	3.28e-6	4.10x	
Tersoff	bulk Si	32000	1 fmsec	16.6	9.2 Mb	3.74e-6	4.67x	
ADP	bulk Ni	32000	5 fmsec	83.6	25 Mb	5.58e-6	6.97x	
EIM	crystalline NaCl	32000	0.5 fmsec	98.9	14 Mb	5.60e-6	6.99x	
Peridynamics	glass fracture	32000	22.2 nsec	422	144 Mb	7.46e-6	9.31x	
SPC/E	liquid water	36000	2 fmsec	700	86 Mb	8.77e-6	11.0x	
CHARMM + PPPM	solvated protein	32000	2 fmsec	376	124 Mb	1.13e-5	14.1x	
MEAM	bulk Ni	32000	5 fmsec	48.8	54 Mb	1.32e-5	16.5x	
Gay-Berne	ellipsoid mixture	32768	0.002 tau	140	21 Mb	2.20e-5	27.5x	
BOP	bulk CdTe	32000	1 fmsec	4.4	74 Mb	2.51e-5	31.3x	
AIREBO	polyethylene	32640	0.5 fmsec	681	101 Mb	3.25e-5	40.6x	
ReaxFF/C	PETN crystal	32480	0.1 fmsec	667	976 Mb	1.09e-4	136x	
COMB	crystalline SiO2	32400	0.2 fmsec	572	85 Mb	2.00e-4	250x	
eFF	H plasma	32000	0.001 fmsec	5066	365 Mb	2.16e-4	270x	

Bond styles (also angle, dihedral, improper)

- Used for molecules with fixed bonds
  - Fix bond/react, bond/create, bond/break can form & break them
- To learn what bond styles LAMMPS has ...
  where would you look?

Bond styles (also angle, dihedral, improper)

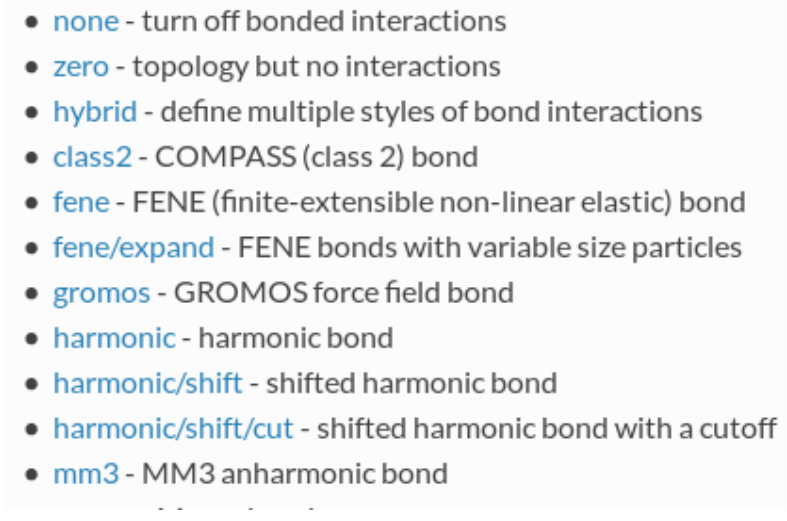
- Used for molecules with fixed bonds
  - Fix bond/react, bond/create, bond/break can form & break them
- To learn what bond styles LAMMPS has ...
  where would you look?
- doc/Commands_bond.html or doc/bond_style.html



5.9. Bond_style potentials

All LAMMPS bond_style commands. Some styles have accelerated versions. This i

none	zero	hybrid	
class2 (ko)	fene (iko)	fene/expand (o)	gromos (o)
harmonic (iko)	harmonic/shift (o)	harmonic/shift/cut (o)	mm3
morse (o)	nonlinear (o)	oxdna/fene	oxdna2/fene
quartic (o)	table (o)		



- none - turn off bonded interactions
- zero - topology but no interactions
- hybrid - define multiple styles of bond interactions
- class2 - COMPASS (class 2) bond
- fene - FENE (finite-extensible non-linear elastic) bond
- fene/expand - FENE bonds with variable size particles
- gromos - GROMOS force field bond
- harmonic - harmonic bond
- harmonic/shift - shifted harmonic bond
- harmonic/shift/cut - shifted harmonic bond with a cutoff
- mm3 - MM3 anharmonic bond

Long-range Coulombics

KSpace style in LAMMPS lingo, see doc/kspace_style.html

- Options:
  - traditional Ewald, scales as $O(N^{3/2})$
  - PPPM (like PME), scales as $O(N \log(N))$
  - MSM, scales as $O(N)$
  - USER-SCAFACOS, wrapper on ScaFaCoS lib
- Additional options:
  - non-periodic, PPPM (z) vs MSM (xyz)
  - long-range dispersion (LJ)

Long-range Coulombics

KSpace style in LAMMPS lingo, see doc/kspace_style.html

- Options:
  - traditional Ewald, scales as $O(N^{3/2})$
  - PPPM (like PME), scales as $O(N \log(N))$
  - MSM, scales as $O(N)$
  - USER-SCAFACOS, wrapper on ScaFaCoS lib
- Additional options:
  - non-periodic, PPPM (z) vs MSM (xyz)
  - long-range dispersion (LJ)
- PPPM is fastest choice for most systems
  - FFTs can scale poorly for large processor counts
- MSM can be faster for low-accuracy or large proc counts
- Pay attention to cutoff & accuracy settings
  - can affect performance dramatically (see timing output)
  - adjusts Real versus KSpace work

Fixes

Most flexible feature in LAMMPS
Allow control of "what" happens "when" within each timestep

Loop over timesteps:

communicate ghost atoms

build neighbor list (once in a while)
compute forces
communicate ghost forces

output to screen and files

Fixes

Most flexible feature in LAMMPS
Allow control of "what" happens "when" within each timestep

Loop over timesteps:

- fix initial — NVE, NVT, NPT, rigid-body integration
- communicate ghost atoms
- fix neighbor — insert particles
- build neighbor list (once in a while)
- compute forces
- communicate ghost forces
- fix force — SHAKE, langevin drag, wall, spring, gravity
- fix final — NVE, NVT, NPT, rigid-body integration
- fix end — volume & T rescaling, diagnostics
- output to screen and files

Fixes

- ∼210 fixes in LAMMPS
- You choose what group of atoms to apply fix to
- Already saw some in obstacle example:
  - fix 1 all nve
  - fix 2 flow temp/rescale 200 1.0 1.0 0.02 1.0
  - fix 3 lower setforce 0.0 0.0 0.0
  - fix 5 upper aveforce 0.0 -0.5 0.0
  - fix 6 flow addforce 1.0 0.0 0.0

Fixes

- ∼210 fixes in LAMMPS
- You choose what group of atoms to apply fix to
- Already saw some in obstacle example:
  - fix 1 all nve
  - fix 2 flow temp/rescale 200 1.0 1.0 0.02 1.0
  - fix 3 lower setforce 0.0 0.0 0.0
  - fix 5 upper aveforce 0.0 -0.5 0.0
  - fix 6 flow addforce 1.0 0.0 0.0
- To learn what fix styles LAMMPS has ...

Fixes

- ~210 fixes in LAMMPS
- You choose what group of atoms to apply fix to
- Already saw some in obstacle example:
  - fix 1 all nve
  - fix 2 flow temp/rescale 200 1.0 1.0 0.02 1.0
  - fix 3 lower setforce 0.0 0.0 0.0
  - fix 5 upper aveforce 0.0 -0.5 0.0
  - fix 6 flow addforce 1.0 0.0 0.0
- To learn what fix styles LAMMPS has ...
  doc/Commands_fix.html or doc/fix.html

Fixes

- ~210 fixes in LAMMPS
- You choose what group of atoms to apply fix to
- Already saw some in obstacle example:
  - fix 1 all nve
  - fix 2 flow temp/rescale 200 1.0 1.0 0.02 1.0
  - fix 3 lower setforce 0.0 0.0 0.0
  - fix 5 upper aveforce 0.0 -0.5 0.0
  - fix 6 flow addforce 1.0 0.0 0.0
- To learn what fix styles LAMMPS has ...
  doc/Commands_fix.html or doc/fix.html
- If you familiarize yourself with fixes,
  you'll know many things LAMMPS can do
- Many fixes store output accessible by other commands
  - thermostat energy, forces on wall, rigid body COMs, etc

Computes

- ∼140 computes in LAMMPS
- Calculate some property of system, in parallel
- Always for the current timestep
- To learn what compute styles LAMMPS has ...

Computes

- ∼140 computes in LAMMPS
- Calculate some property of system, in parallel
- Always for the current timestep
- To learn what compute styles LAMMPS has ...
  doc/Commands_compute.html or doc/compute.html

Computes

- ∼140 computes in LAMMPS
- Calculate some property of system, in parallel
- Always for the current timestep
- To learn what compute styles LAMMPS has ...
  doc/Commands_compute.html or doc/compute.html

5.7. Compute commands

An alphabetic list of all LAMMPS compute commands. Some styles have accelerated versions. This is indicated by additional letters in parenthesis:

ackland/atom	adf	aggregate/atom	angle	angle/local	angmom/chunk
basal/atom	body/local	bond	bond/local	centro/atom	chunk/atom
chunk/spread/atom	cluster/atom	cna/atom	cnp/atom	com	com/chunk
contact/atom	coord/atom	damage/atom	dihedral	dihedral/local	dilatation/atom
dipole/chunk	displace/atom	dpd	dpd/atom	edpd/temp/atom	entropy/atom
erotate/asphere	erotate/rigid	erotate/sphere	erotate/sphere/atom	event/displace	fep
force/tally	fragment/atom	global/atom	group/group	gyration	gyration/chunk
gyration/shape	heat/flux	heat/flux/tally	hexorder/atom	improper	improper/local
inertia/chunk	ke	ke/atom	ke/atom/eff	ke/eff	ke/rigid
meso/e/atom	meso/rho/atom	meso/t/atom	momentum	msd	msd/chunk

Computes

- Key point:
  - computes store results of their calculations
  - other commands invoke them and use the results
  - e.g. thermo output, dumps, fixes
- Output of computes:
  - global vs per-atom vs local
  - scalar vs vector vs array
  - extensive vs intensive values

Computes

- Key point:
  - computes store results of their calculations
  - other commands invoke them and use the results
  - e.g. thermo output, dumps, fixes
- Output of computes:
  - global vs per-atom vs local
  - scalar vs vector vs array
  - extensive vs intensive values
- Examples:
  - temp & pressure = global scalar or vector
  - pe/atom = potential energy per atom (vector)
  - displace/atom = displacement per atom (array)
  - pair/local & bond/local = per-neighbor or per-bond info
- Many computes are useful with averaging fixes:
  - fix ave/time, ave/chunk (spatial), ave/atom, ave/histo, ave/correlate

Thermo output

One line of output every N timesteps to screen and log file

- See doc/thermo_style.html

Thermo output

One line of output every N timesteps to screen and log file

- See doc/thermo_style.html
- Any scalar can be output:
  - dozens of keywords: temp, pyy, eangle, lz, cpu
  - any output of a compute or fix: c_ID, f_ID[N], c_ID[N][M]
    - fix ave/time stores time-averaged quantities
  - equal-style variable: v_MyVar
  - one value from atom-style variable: v_xx[N]
  - any property for one atom: q, fx, quat, etc

Thermo output

One line of output every N timesteps to screen and log file

- See doc/thermo_style.html
- Any scalar can be output:
  - dozens of keywords: temp, pyy, eangle, lz, cpu
  - any output of a compute or fix: c_ID, f_ID[N], c_ID[N][M]
    - fix ave/time stores time-averaged quantities
  - equal-style variable: v_MyVar
  - one value from atom-style variable: v_xx[N]
  - any property for one atom: q, fx, quat, etc
- Post-process via:
  - tools/python/logplot.py log.lammps X Y (via GnuPlot)
  - tools/python/log2txt.py log.lammps data.txt X Y ...
  - they can read thermo output across multiple runs

Dump output

Snapshot of per-atom values every N timesteps

- See doc/dump.html

Dump output

Snapshot of per-atom values every N timesteps

- See doc/dump.html
- Styles:
  - atom, custom (both native LAMMPS)
    - VMD will auto-read if file named *.lammpstraj
  - xyz for coords only
  - cfg for AtomEye
  - DCD, XTC for CHARMM, NAMD, GROMACS
    - good for back-and-forth runs and analysis
  - HDF5, NetCDF, VTK, AIDIOS, molfile
- Additional styles:
  - local: per-neighbor, per-bond, etc info
  - image: instant picture, rendered in parallel
  - movie: instant movie, rendered in parallel

Dump output

- Any per-atom quantity can be output
  - dozens of keywords: id, type, x, xs, xu, mux, omegax, ...
  - any output of a compute or fix: f_ID, c_ID[M]
  - atom-style variable: v_foo

Dump output

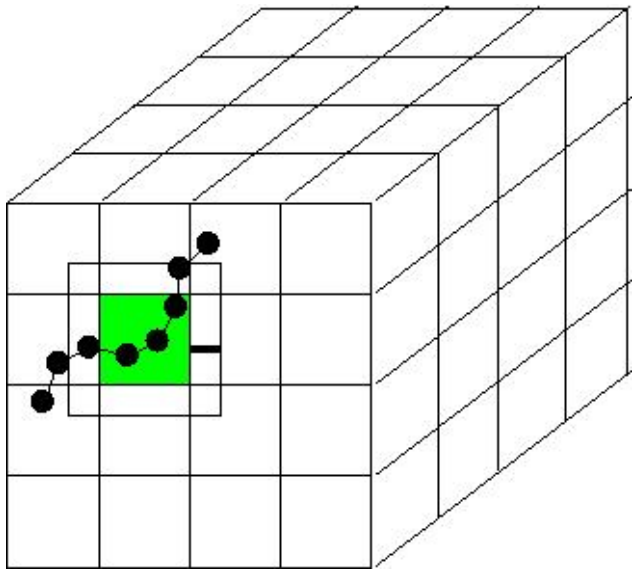
- Any per-atom quantity can be output
  - dozens of keywords: id, type, x, xs, xu, mux, omegax, ...
  - any output of a compute or fix: f_ID, c_ID[M]
  - atom-style variable: v_foo
- Additional options:
  - control which atoms by group or region
  - control which atoms by threshold
    - dump_modify thresh c_pe $>$ 3.0
  - text or binary or gzipped
  - one big file or per snapshot
  - parallel output: per-processor or N files or MPIIO package

Dump output

- Any per-atom quantity can be output
  - dozens of keywords: id, type, x, xs, xu, mux, omegax, ...
  - any output of a compute or fix: f_ID, c_ID[M]
  - atom-style variable: v_foo
- Additional options:
  - control which atoms by group or region
  - control which atoms by threshold
    - dump_modify thresh c_pe $>$ 3.0
  - text or binary or gzipped
  - one big file or per snapshot
  - parallel output: per-processor or N files or MPIIO package
- Post-run conversion
  - tools/python/dump2cfg.py, dump2pdb.py, dump2xyz.py
  - Pizza.py dump, cfg, ensight, pdb, svg, vtk, xyz

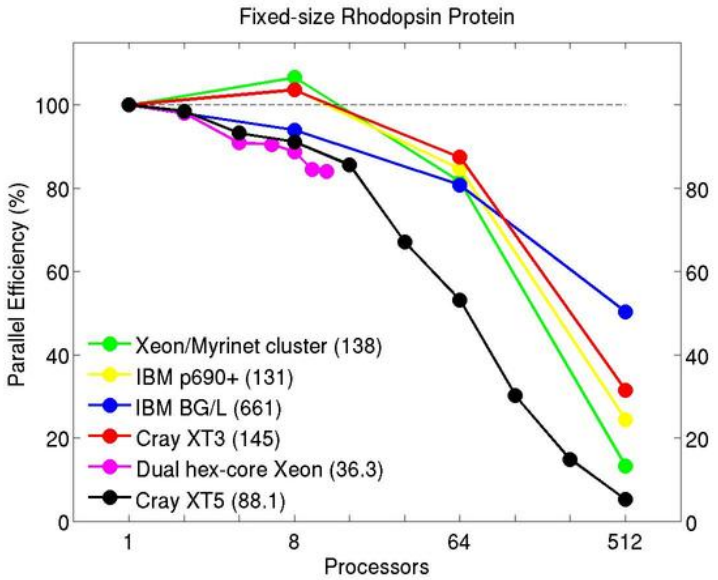
Parallelization in LAMMPS

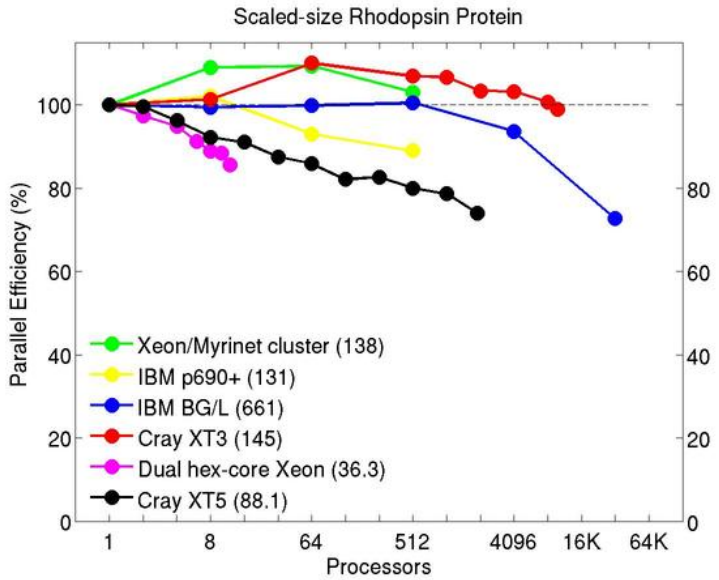
- Physical domain divided into 3d bricks
- One brick per processor
- Atoms carry properties & topology as they migrate
- Comm of ghost atoms within cutoff
  - 6-way local stencil
- Short-range forces $\Rightarrow$ CPU cost scales as $O(N/P)$



Parallel performance

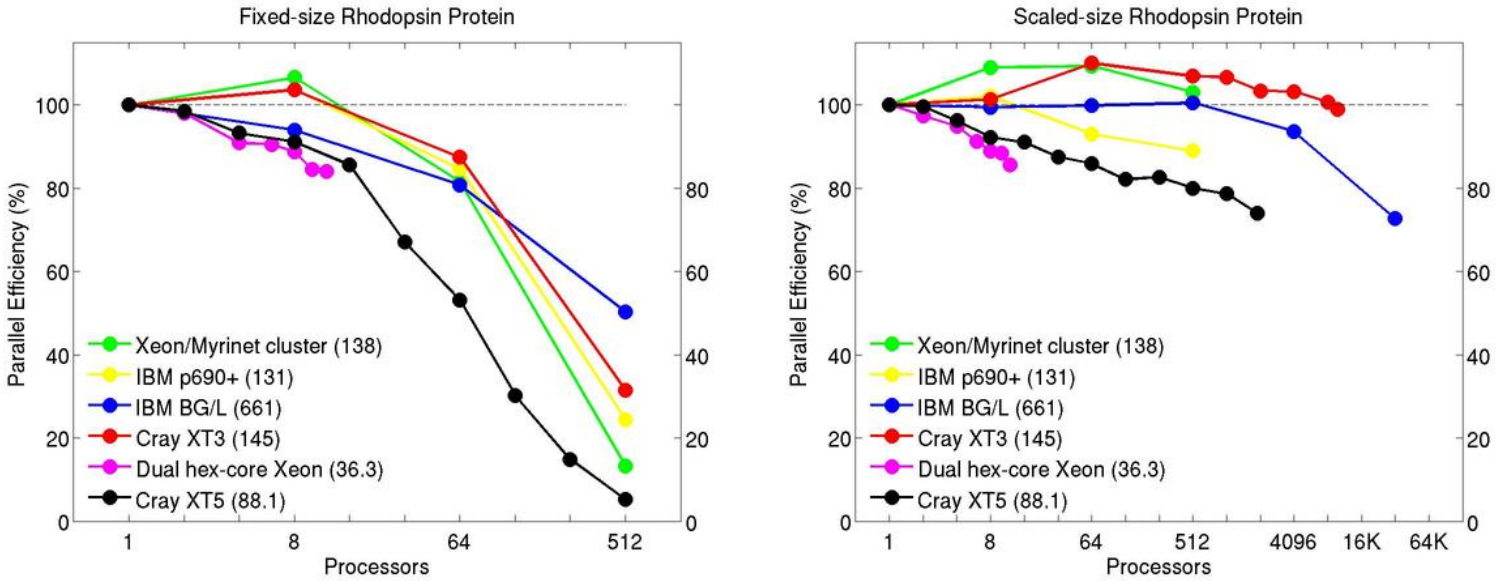
See http://lammps.sandia.gov/bench.html
Strong or weak scaling, O(N/P) until too few atoms/proc





Parallel performance

See http://lammps.sandia.gov/bench.html
Strong or weak scaling, O(N/P) until too few atoms/proc



Exercise: run bench/in.lj, change N and P, is it $O(N/P)$?

- lmp_linux -v x 2 -v y 2 -v z 2 < in.lj # change N
- mpirun -np P lmp_linux < in.lj # change P

How to speed-up your simulations

See doc/Speed.html of manual

1. Many ideas for long-range Coulombics
   - PPPM with 2 vs 4 FFTs (smoothed PPPM)
   - PPPM with staggered grid
   - run_style verlet/split
   - processor layout

How to speed-up your simulations

See doc/Speed.html of manual

1. Many ideas for long-range Coulombics
   - PPPM with 2 vs 4 FFTs (smoothed PPPM)
   - PPPM with staggered grid
   - run_style verlet/split
   - processor layout
2. OPT, GPU, USER-INTEL, USER-OMP, KOKKOS packages
   - OPT for CPU
   - GPU for NVIDIA GPUs with multiple cores/GPU
   - USER-INTEL for Intel CPU and KNL
   - USER-OMP for OpenMP on multicore CPUs
   - KOKKOS for OpenMP, KNL, GPU

How to speed-up your simulations (2)

3. Increase time scale via timestep size
   - fix shake for rigid bonds (2 fs)
   - run_style respa for hierarchical timesteps (4 fs)
   - hyper command for rare-event systems

4. Increase length scale via coarse graining
   - all-atom vs united-atom vs bead-spring
   - mesoscale models:
     ASPHERE, BODY, COLLOID, FLD packages
     GRANULAR, PERI, RIGID, SRD packages
   - see doc/Packages.html for details

Quick tour of more advanced topics

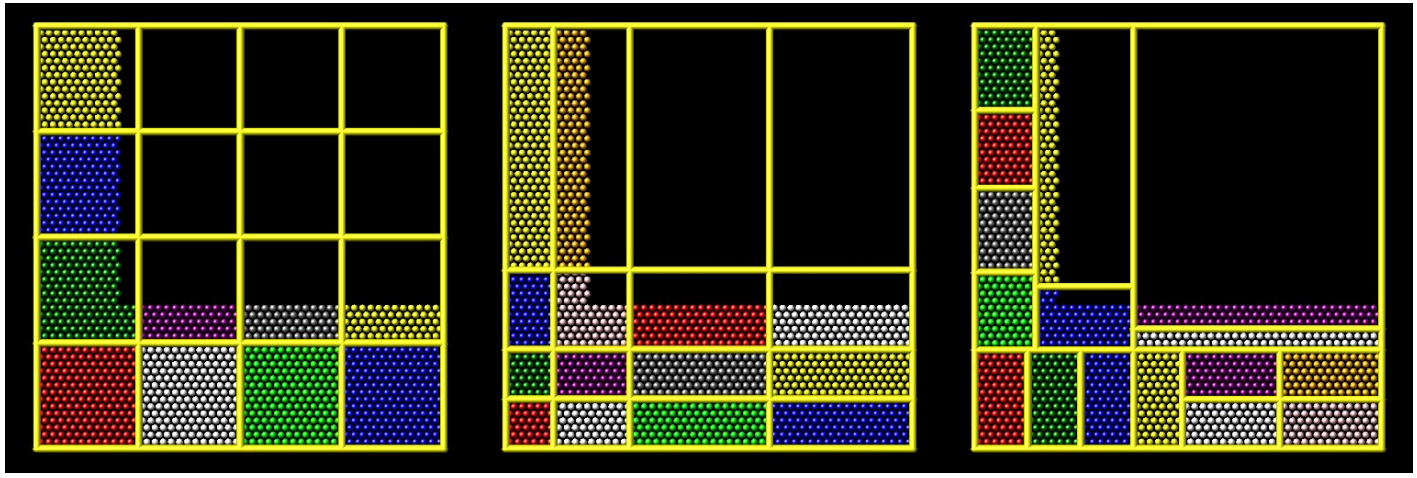
- Units
  - see doc/units.html
  - LJ, real, metal, cgs, si
  - all input/output in one unit system
- Ensembles
  - see doc/Howto_thermostat.html, doc/Howto_barostat.html
  - one or more thermostats (by group)
  - single barostat
  - rigid body dynamics
- Hybrid models
  - pair_style hybrid and hybrid/overlay
  - atom_style hybrid sphere bond ...

Quick tour of more advanced topics (2)

- Aspherical particles
  - see doc/Howto_spherical.html
  - ellipsoidal, lines, triangles, rigid bodies
  - ASPHERE package
- Mesoscale and continuum models
  - Corase-grained packages mentioned 2 slides ago
  - PERI package for Peridynamics
  - USER-ATC package for atom-to-continuum (FE)
  - USER-SPH, USER-SMD packages for smoothed particle hydro
  - GRANULAR package for granular media
  - LIGGGHTS simulator for DEM (external code)
    - www.liggghts.com/www.cfdem.com
    - built on top of LAMMPS

Quick tour of more advanced topics (3)

- Multi-replica modeling
  - REPLICA package, see doc/Howto_replica.html
  - parallel tempering, PRD, TAD, NEB
- Load balancing
  - balance command for static LB
  - fix balance command for dynamic LB
  - adjusting proc dividers, or recursive coordinate bisection
  - weighted balancing now an option



Quick tour of more advanced topics (4)

- Energy minimization
- Via usual dynamics
  - pair_style soft
  - fix nve/limit and fix viscous
- Via gradient-based minimization
  - min_style cg, htfn, sd
- Via damped-dynamics minimization
  - min_style quickmin and fire
  - used for nudged-elastic band (NEB)

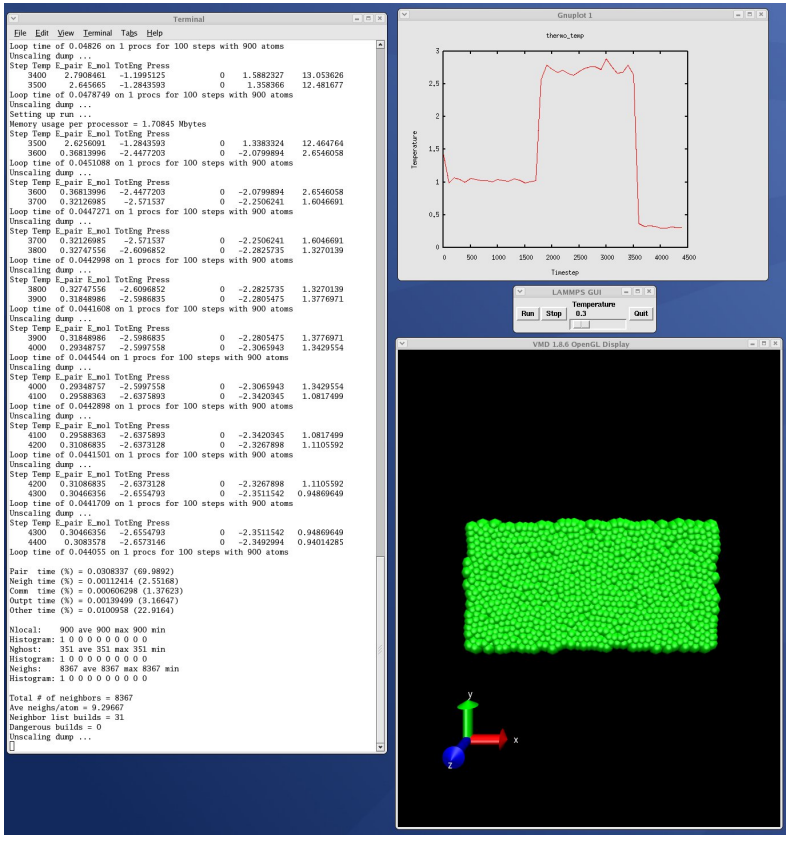
Quick tour of more advanced topics (4)

- Energy minimization
- Via usual dynamics
  - pair_style soft
  - fix nve/limit and fix viscous
- Via gradient-based minimization
  - min_style cg, htfn, sd
- Via damped-dynamics minimization
  - min_style quickmin and fire
  - used for nudged-elastic band (NEB)
- Packages, packages, packages ...
  - package = collection of commands with a theme, or wrapper on an external or provided library
  - ∼70 packages in LAMMPS, wide variety
  - see doc/Packages.html for details

Quick tour of more advanced topics (5)

Use LAMMPS as a library and from Python

- doc/Howto_library.html, doc/Python_head.html
- lib has C-style interface (C, C++, Fortran, Python)
- examples/COUPLE dir
- python and python/examples directories



Customizing and modifying LAMMPS

- 95% of LAMMPS is customized add-on classes, via styles
- Write a new derived class, drop into src, re-compile

Customizing and modifying LAMMPS

- 95% of LAMMPS is customized add-on classes, via styles
- Write a new derived class, drop into src, re-compile
- Resources:
  - doc/Modify.html
  - doc/PDF/Developer.pdf
    - class hierarchy & timestep structure
  - Workshops and Tutorial links on LAMMPS web site:
    - slides for hackers/developers breakout of past workshops
- Adding features to LAMMPS breakout session, Wed 2:15 PM
- Please contribute your new code to the LAMMPS
  - doc/Modify_contribute.html:
    Submitting new features for inclusion in LAMMPS
- GitHub site: https://github.com/lammps/lammps

What have people already done with LAMMPS?

- Pictures: http://lammps.sandia.gov/pictures.html
- Movies: http://lammps.sandia.gov/movies.html

	evaporation self-assembly		Compression of nanoparticles
	GCMC model of zeolite occupancy		self-assembling nanofibers from Thiophene-peptide oligomers
	layer-by-layer self assembly		smoothed particle hydrodynamics (SPH) models
	dislocations moving thru grain boundaries		electron force field for non-adiabatic dynamics
	granular particles flowing from hopper		granular Discrete Elememt Method (DEM) models
	fiber dynamics		brazing of two-metal system
	Peridynamics mesoscale modeling of impact fracture		shear faults in a model brittle solid
	stick/slip and polymer flow on rough surfaces		crystallization of polyethylene melt
	melting of polycrystalline metal		deformation and void nucleation under shock loading
	dynamics of an isolated edge dislocation		cavitation in liquid metal
	nanoprecipitates and shock induced plasticity		Brazil nut effect
	ultra-thin Cu nanowire formation		Cu nanowire loading and unloading
	Au nanowire formation and extension		flow of water and ions thru a silica pore
	metal response to He bubble formation		dynamics of rhodopsin protein in lipid membrane
	CO2 escaping from binding pocket of RuBisCO protein		C-terminus of RuBisCO closing over binding pocket
	entropy-driven nano-motor		metal solidification
	liquid crystal conformations		

What have people already done with LAMMPS?

- Pictures: http://lammps.sandia.gov/pictures.html
- Movies: http://lammps.sandia.gov/movies.html

	evaporation self-assembly		Compression of nanoparticles
	GCMC model of zeolite occupancy		self-assembling nanofibers from Thiophene-peptide oligomers
	layer-by-layer self assembly		smoothed particle hydrodynamics (SPH) models
	dislocations moving thru grain boundaries		electron force field for non-adiabatic dynamics
	granular particles flowing from hopper		granular Discrete Elememt Method (DEM) models
	fiber dynamics		brazing of two-metal system
	Peridynamics mesoscale modeling of impact fracture		shear faults in a model brittle solid
	stick/slip and polymer flow on rough surfaces		crystallization of polyethylene melt
	melting of polycrystalline metal		deformation and void nucleation under shock loading
	dynamics of an isolated edge dislocation		cavitation in liquid metal
	nanoprecipitates and shock induced plasticity		Brazil nut effect
	ultra-thin Cu nanowire formation		Cu nanowire loading and unloading
	Au nanowire formation and extension		flow of water and ions thru a silica pore
	metal response to He bubble formation		dynamics of rhodopsin protein in lipid membrane
	CO2 escaping from binding pocket of RuBisCO protein		C-terminus of RuBisCO closing over binding pocket
	entropy-driven nano-motor		metal solidification
	liquid crystal conformations		

- Papers: http://lammps.sandia.gov/papers.html
  - authors, titles, abstracts for 1000s of papers

Extra provided document

- lammps_examples_Aug19.pdf
- Included in your virtual image
- Can use today, if you wish, to explore/modify example scripts included in LAMMPS distro
- Explanation and ideas for 7 example scripts

Exercises with the LAMMPS examples

examples/README has one-line descriptions of 40 examples

Quick runs (2d) and visually appealing:

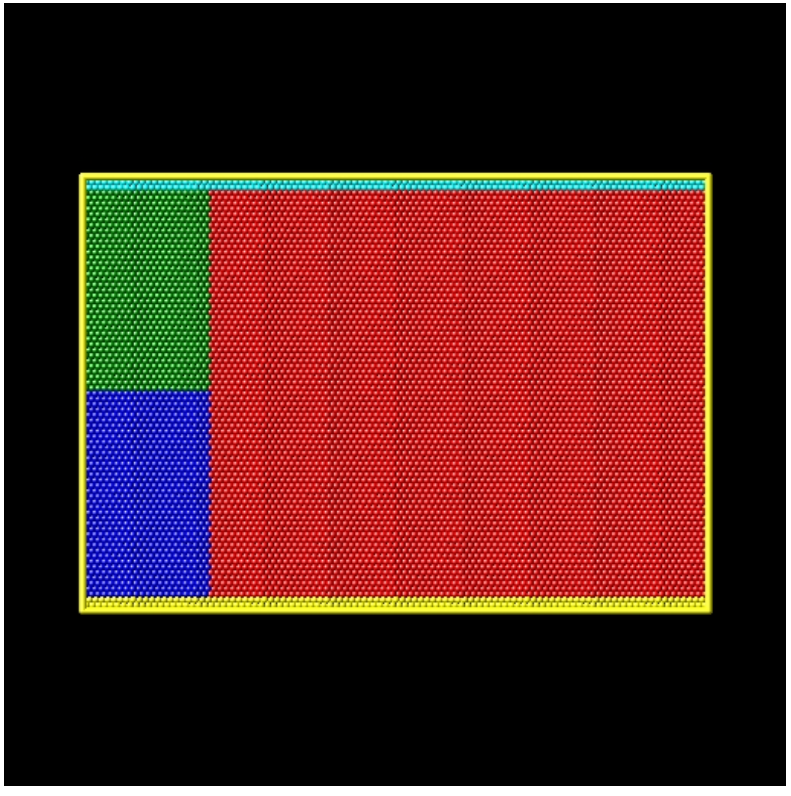
- crack: crack propagation
- flow: Couette and Poiseuille flow in a channel
- friction: frictional contact of spherical asperities
- indent: spherical indenter into solid
- micelle: self-assembly of small lipid-like molecules
- obstacle: flow around two voids in a channel
- shear: sideways shear of solid, with and without a void

Running and visualizing the examples

- Run in serial
  - lmp_linux < in.friction
- Run in parallel
  - mpirun -np 4 lmp_linux < in.friction
- Uncomment dump image and dump_modify lines
  - produce series of JPG (or PPM) files
- Uncomment dump atom line
  - produce snapshot file, can viz with VMD

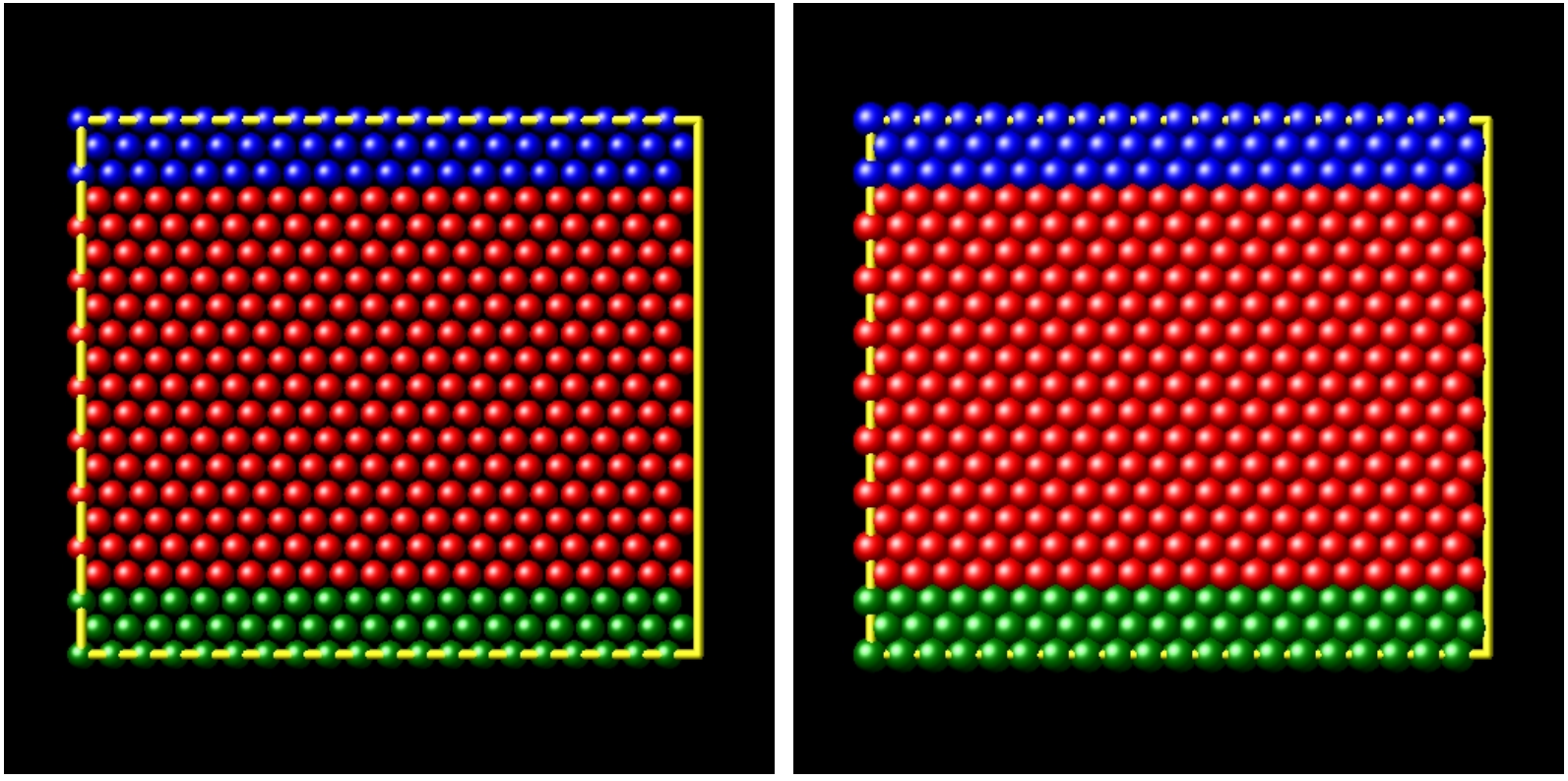
Crack problem

- Tensile pull on 2d LJ solid
- Slit crack between red/green
  neigh_modify exclude 2 3
- Uniform gradient pull
  velocity ramp command
  else shock waves or worse
- Need large system & slow pull
  else defects besides crack
- Options to play with:
  pull rate
  pair-wise cutoff
  turn off velocity ramp
  change NULL $\Rightarrow$ 0.0 in fix 2



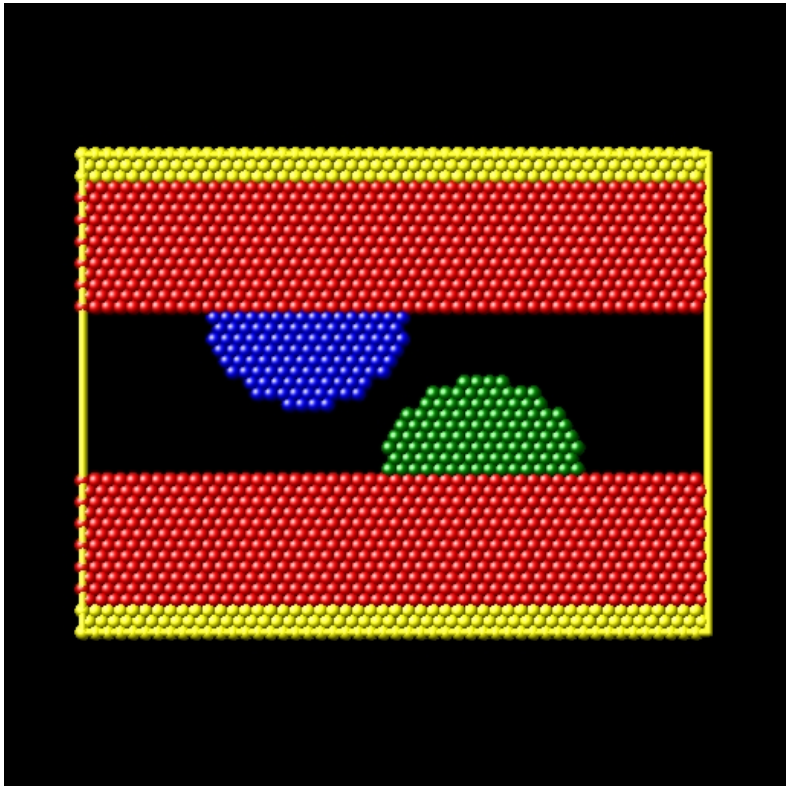
Flow problems

- Couette flow and Poiseuille flow
- Options to play with: wall velocity, force kick, temperature
- Monitor velocity profile via fix ave/chunk or spatial



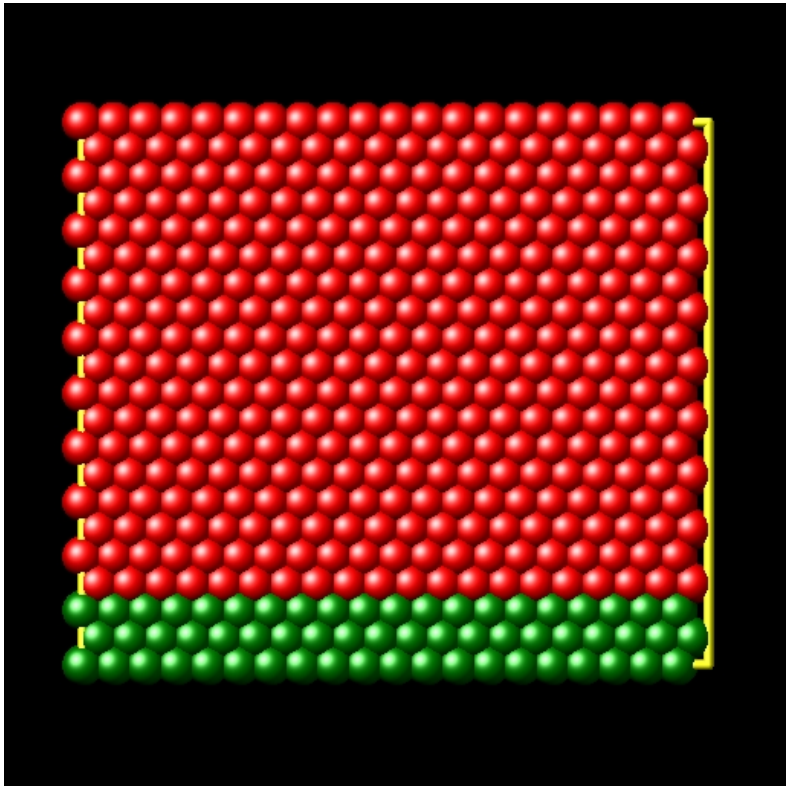
Friction problem

- 2 non-planar surfaces
- Region commands
  to build geometry
- Options to play with:
  asperity size, shape
  asperity separation
  x-velocity
  multiple passes



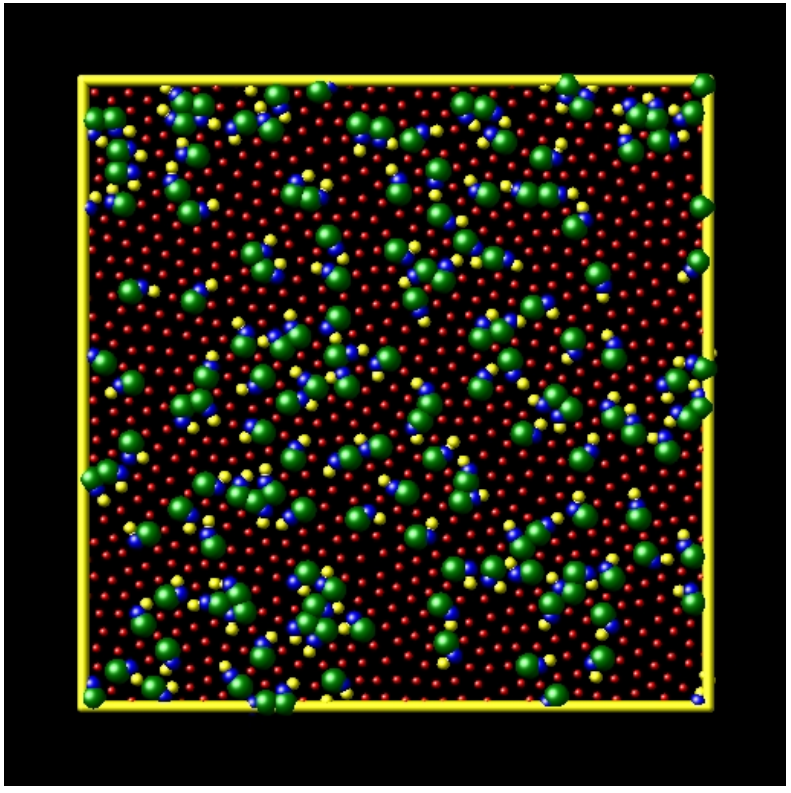
Indent problem

- 2d LJ solid
  - periodic in x
  - free upper y surface
- Spherical indenter
  - downward push, remove
- Defect creation & healing
- Options to play with:
  - speed & depth of indent
  - size of indenter
  - size of system



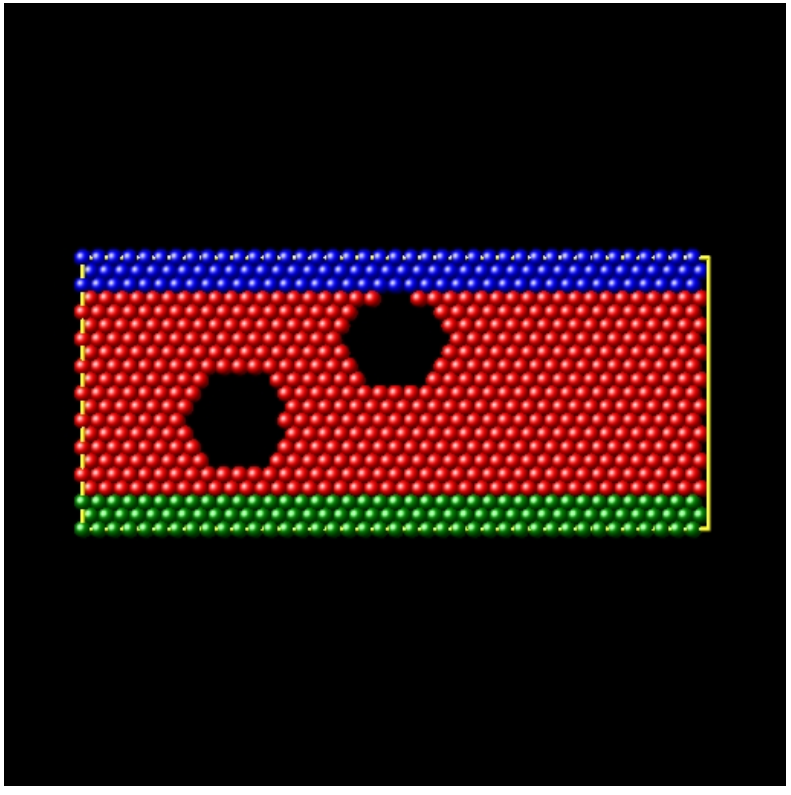
Micelle problem

- Simple lipid model
  - hydrophilic head
  - hydrophobic tail
  - monomer solvent
- 2d self-assembly
  - vesicles, bilayers
- Options to play with:
  - timestep size
  - # of timesteps
  - pair-wise coeffs



Obstacle problem

- LJ flow around obstacle(s)
- Poiseuille kick added to atoms
  pressure-gradient flow
- Top surface applies pressure
- Obstacle creation
  delete_atoms command
  fix indent command
- Options to play with:
  size of force kick
  size of system
  size & position of obstacles
  shape of obstacles
  add a new obstacle



Shear problems

- Fixed-end shear in fcc Ni
- EAM potential
- Quasi-3d
  - non-periodic XY slab
  - thin in Z, periodic
- Defect formation without and with void
- Options to play with:
  - size of system
  - shear rate
  - turn off velocity ramp
  - change void shape, size
  - add another void