24-623: LAMMPS Lecture 3/8/2011

Outline

- LAMMPS Overview
- LAMMPS Installation
- LAMMPS Example How to Run
- LAMMPS Example Code
- LAMMPS Example Results
- LAMMPS Performance
- LAMMPS Modifications
- LAMMPS Post-Processing
- Discussion

LAMMPS Overview

- LAMMPS = “Large-scale Atomic/Molecular Massively Parallel Simulator”

http://lammps.sandia.gov/

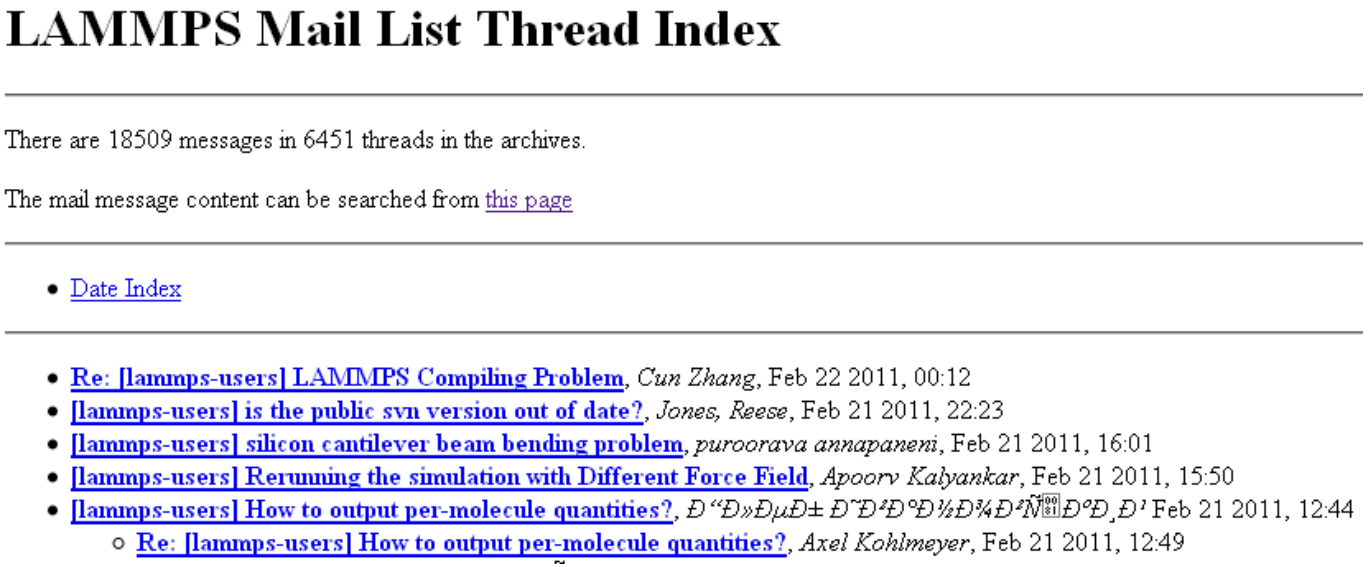
FEATURES:

- **LAMMPS has many potentials**, including potentials for biological systems, colloids, granular materials, metals, etc.
- **LAMMPS has many built in functions** (computes) that can automatically calculate/average quantities (Temp., Press., MSD of atoms, RDF, thermal conductivity, etc.).
- **LAMMPS is an open source code** in C++. Anyone can modify the code. Older versions in F90/F77 are available as well as Windows version.
- **LAMMPS is a parallel code**. It can be run on single cpu or in parallel using MPI and spatial-decomposition of simulation domain.

LAMMPS Overview: Support

- LAMMPS has an excellent mailing list that began in 2005 and has >10,000 messages. Probably the best discussion forum I've seen of all open source simulation codes.

http://lammps.sandia.gov/mail.html



LAMMPS Mail List Thread Index

There are 18509 messages in 6451 threads in the archives.

The mail message content can be searched from this page

- Date Index

- Re: [lammps-users] LAMMPS Compiling Problem, *Cun Zhang*, Feb 22 2011, 00:12
- [lammps-users] is the public svn version out of date?, *Jones, Reese*, Feb 21 2011, 22:23
- [lammps-users] silicon cantilever beam bending problem, *puroorava annapaneni*, Feb 21 2011, 16:01
- [lammps-users] Rerunning the simulation with Different Force Field, *Apoorv Kalyankar*, Feb 21 2011, 15:50
- [lammps-users] How to output per-molecule quantities?, Đ "Đ»ĐµĐ± ĐˆĐ²Đ°Đ½Đ¾Đ²Ñ‹Đ°Đ¸Đ¹ Feb 21 2011, 12:44
  - Re: [lammps-users] How to output per-molecule quantities?, *Axel Kohlmeyer*, Feb 21 2011, 12:49

Major Contributors:

- Steve Plimpton: staff member at Sandia Nat. Lab. http://www.sandia.gov/~sjplimp/
- Axel Kohlmeyer: Professor at Temple Univ. http://sites.google.com/site/akohlmey/
- You!

Mechanical Engineering

LAMMPS Overview: Potentials

- pair_style none - turn off pairwise interactions
- pair_style hybrid - multiple styles of pairwise interactions
- pair_style hybrid/overlay - multiple styles of superposed pairwise interactions

- pair_style airebo - AI-REBO potential
- pair_style born - Born-Mayer-Huggins potential
- pair_style born/coul/long - Born-Mayer-Huggins with long-range Coulomb
- pair_style buck - Buckingham potential
- pair_style buck/coul/cut - Buckingham with cutoff Coulomb
- pair_style buck/coul/long - Buckingham with long-range Coulomb
- pair_style colloid - integrated colloidal potential
- pair_style coul/cut - cutoff Coulombic potential
- pair_style coul/debye - cutoff Coulombic potential with Debye screening
- pair_style coul/long - long-range Coulombic potential
- pair_style dipole/cut - point dipoles with cutoff
- pair_style dpd - dissipative particle dynamics (DPD)
- pair_style dpd/tstat - DPD thermostatting
- pair_style dsmc - Direct Simulation Monte Carlo (DSMC)
- pair_style eam - embedded atom method (EAM)
- pair_style eam/opt - optimized version of EAM
- pair_style eam/alloy - alloy EAM
- pair_style eam/alloy/opt - optimized version of alloy EAM
- pair_style eam/fs - Finnis-Sinclair EAM
- pair_style eam/fs/opt - optimized version of Finnis-Sinclair EAM
- pair_style eim - embedded ion method (EIM)
- pair_style gauss - Gaussian potential
- pair_style gayberne - Gay-Berne ellipsoidal potential
- pair_style gayberne/gpu - GPU-enabled Gay-Berne ellipsoidal potential

- pair_style lubricate - hydrodynamic lubrication forces
- pair_style meam - modified embedded atom method (MEAM)
- pair_style morse - Morse potential
- pair_style morse/opt - optimized version of Morse potential
- pair_style peri/lps - peridynamic LPS potential
- pair_style peri/pmb - peridynamic PMB potential
- pair_style reax - ReaxFF potential
- pair_style resquared - Everaers RE-Squared ellipsoidal potential
- pair_style soft - Soft (cosine) potential
- pair_style sw - Stillinger-Weber 3-body potential
- pair_style table - tabulated pair potential
- pair_style tersoff - Tersoff 3-body potential
- pair_style tersoff/zbl - Tersoff/ZBL 3-body potential
- pair_style yukawa - Yukawa potential
- pair_style yukawa/colloid - screened Yukawa potential for finite-size particles

- pair_style gran/hertz/history - granular potential with Hertzian interactions
- pair_style gran/hooke - granular potential with history effects
- pair_style gran/hooke/history - granular potential without history effects
- pair_style hbond/dreiding/lj - DREIDING hydrogen bonding LJ potential
- pair_style hbond/dreiding/morse - DREIDING hydrogen bonding Morse potential
- pair_style lj/charmm/coul/charmm - CHARMM potential with cutoff Coulomb
- pair_style lj/charmm/coul/charmm/implicit - CHARMM for implicit solvent
- pair_style lj/charmm/coul/long - CHARMM with long-range Coulomb
- pair_style lj/charmm/coul/long/gpu - GPU-enabled version of CHARMM with long-range Coulomb
- pair_style lj/charmm/coul/long/opt - optimized version of CHARMM with long-range Coulomb
- pair_style lj/class2 - COMPASS (class 2) force field with no Coulomb
- pair_style lj/class2/coul/cut - COMPASS with cutoff Coulomb
- pair_style lj/class2/coul/long - COMPASS with long-range Coulomb
- pair_style lj/cut - cutoff Lennard-Jones potential with no Coulomb
- pair_style lj/cut/gpu - GPU-enabled version of cutoff LJ
- pair_style lj/cut/opt - optimized version of cutoff LJ
- pair_style lj/cut/coul/cut - LJ with cutoff Coulomb
- pair_style lj/cut/coul/cut/gpu - GPU-enabled version of LJ with cutoff Coulomb
- pair_style lj/cut/coul/debye - LJ with Debye screening added to Coulomb
- pair_style lj/cut/coul/long - LJ with long-range Coulomb
- pair_style lj/cut/coul/long/gpu - GPU-enabled version of LJ with long-range Coulomb
- pair_style lj/cut/coul/long/tip4p - LJ with long-range Coulomb for TIP4P water
- pair_style lj/expand - Lennard-Jones for variable size particles
- pair_style lj/gromacs - GROMACS-style Lennard-Jones potential
- pair_style lj/gromacs/coul/gromacs - GROMACS-style LJ and Coulombic potential
- pair_style lj/smooth - smoothed Lennard-Jones potential
- pair_style lj96/cut - Lennard-Jones 9/6 potential
- pair_style lj96/cut/gpu - GPU-enabled version of Lennard-Jones 9/6

-Most of these can be combined in a “hybrid” style
- Additionally, users can enter in potentials numerically or by writing a new module to LAMMPS

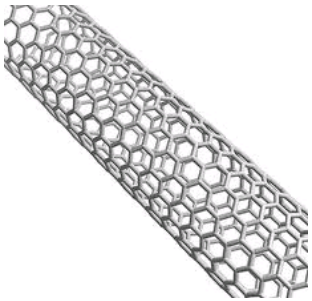
LAMMPS Overview: Potentials 2

FEATURES:

- Potentials for “soft” materials (biomolecules, polymers):

- Can simulate Colloids, bonded organic molecules, granular materials, etc.

- Potentials for “hard” materials (CNTs, Si, Metals, LJ, Charge):

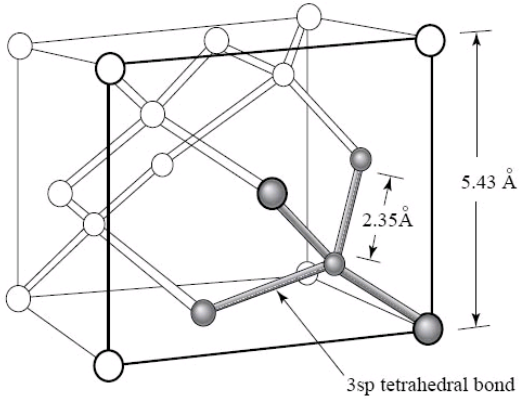


pair_style airebo - AI-REBO potential

Can simulate CNTs/hydrocarbons:

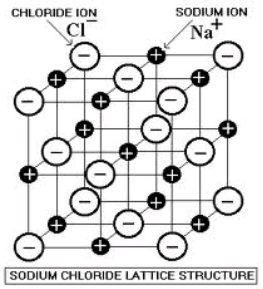
pair_style sw - Stillinger-Weber 3-body potential

Can simulate Si/Ge:



pair_style coul/long - long-range Coulombic potential

Can handle Coulomb, from Class:



$$\Phi = \frac{1}{2}\sum_{i}\sum_{i\neq j}\phi(r_{ij}) \qquad \Phi = \int_{R}^{\infty} 4\pi r^2 \phi(r) dr \rightarrow r^{-2} \rightarrow \int_{R}^{\infty} r^0 dr = \infty$$

Mechanical Engineering

LAMMPS Installation

- LAMMPS can be downloaded from:

  http://lammps.sandia.gov/download.html

- Includes the current "build" (with both serial and parallel capabilities) and older builds.
- Also available are **Windows** serial and parallel versions:

- LAMMPS can be "installed" in **Windows** using Cygwin: http://www.cygwin.com/

Linux

- The current "build" has the most recent functionality. Consequently, it may be incompatible with old compilers/libraries.

Needed:

Serial

- **Desktop PC running (almost) any Linux OS.**
- **C++ compiler (GNU gcc or Intel icc)**: GNU installed by default in almost every Linux OS.

Parallel

- **Parallel computer (Cluster, GPU, etc.).**
- **MPI library (openmpi, impi, mpich) for parallel code**: Mpich and openmpi installed by default on most computing clusters.

LAMMPS Installation 2

- Detailed instructions here:

http://lammps.sandia.gov/doc/Section_start.html

```
root@jason-desktop:/home/jason/lammps# ls
Johan            lammps-18Aug10.tar.gz  lammps.tar.gz  work
lammps-16Apr10   lammps-25Oct10         LJ
lammps-18Aug10   lammps2.tar.gz         log.lammps
```

root@jason-desktop:/home/jason/lammps# tar xvf lammps-18Aug10.tar.gz

- This unzips the LAMMPS program files to a folder called “lammps-18Aug10”

root@jason-desktop:/home/jason/lammps# cd ./lammps-18Aug10/src

- This moves to the “src” folder which has the source code files. Once here, there are a few options. To compile the code type:

root@jason-desktop:/home/jason/lammps/lammps-18Aug10/src# make machine

Where **machine** can be:

```
# mingw = Windows 32bit, cross-compiled on Linux, gcc-4.4.1, MinGW x-compiler
# mkl = Intel Cluster Tools, mpiicc, MKL MPI, MKL FFT
# odin = 1400 cluster, g++, MPICH, no FFTs
# openmpi = Fedora Core 6, mpic++, OpenMPI-1.1, FFTW2
# pgi = Portland Group compiler, pgCC, MPICH, FFTW
# power5 = IBM Power5+, mpCC_r, native MPI, FFTW
# qed = CSRI cluster, mpiCC, MPICH, no FFTs
# redsky - SUN X6275 with dual socket/quad core nodes, mpic++, openmpi, FFTW
# sdsc = SDSC BG/L machine, xlC, native MPI, FFTW
# seaborg = NERSC IBM machine, mpCC, native MPI, FFTW
# serial = RedHat Linux box, g++4, gfortran, no MPI, no FFTs
# serial_debug = RedHat Linux box, g++4, gfortran,  no MPI, no FFTs
# sgi = SGI Origin 350 64-bit, SGI MIPSpro CC, SGI MPI, SGI SCSL MP FFTs
# solaris = Sun box, c++, no MPI, no FFTs
# spirit = HP cluster 64-bit, mpicxx, native MPI, FFTW
# storm = Cray Red Storm XT3, Cray CC, native MPI, FFTW
# tacc = UT Lonestar TACC machine, mpiCC, MPI, FFTW
# tbird = Dell cluster with Xeons, Intel mpicxx, native MPI, FFTW
# tesla = 16-proc SGI Onyx3, g++, no MPI, SGI FFTs
# tunnison - 64-bit dual-core Linux cluster, mpic++, OpenMPI-1.1, FFTW2
# xt3 = PSC BigBen Cray XT3, CC, native MPI, FFTW
```

```
# altix = SGI Altix, Intel icc, MPI, FFTs from SGI SCSL library
# bgl = LLNL Blue Gene Light machine, xlC, native MPI, FFTW
# cygwin = Windows Cygwin, mpicxx, MPICH, FFTW
# encanto = NM cluster with dual quad-core Xeons, mpicxx, native MPI, FFTW
# fink = Mac OS-X w/ fink libraries, c++, no MPI, FFTW 2.1.5
# g++ = RedHat Linux box, g++4, gfortran, MPICH2, FFTW
# g++3 = RedHat Linux box, g++ (v3), Intel ifort, MPICH2, FFTW
# glory = Linux cluster with 4-way quad cores, Intel mpicxx, native MPI, FFTW
# jaguar = ORNL Jaguar Cray XT5, CC, native MPICH, FFTW
# lam = FreeBSD box, mpic++, Intel ifort, LAM/MPI, FFTW
# linux = RedHat Linux box, Intel icc, Intel ifort, MPICH2, FFTW
# mac = Apple PowerBook G4 laptop, c++, Intel ifort, no MPI, FFTW 2.1.5
# mac_mpi = Apple PowerBook G4 laptop, mpic++, gfortran, fink LAM/MPI, fink FFTW
 2.1.5
```

LAMMPS Installation 3

- The previous section will make LAMMPS with the standard “packages”. Additional packages can be included/excluded during installation, but may require additional libraries (fft, etc.).

asphere	aspherical particles and force fields
class2	class 2 force fields
colloid	colloidal particle force fields
dipole	point dipole particles and force fields
dsmc	Direct Simulation Monte Carlo (DMSC) pair style
gpu	GPU-enabled force field styles
granular	force fields and boundary conditions for granular systems
kspace	long-range Ewald and particle-mesh (PPPM) solvers
manybody	metal, 3-body, bond-order potentials
meam	modified embedded atom method (MEAM) potential
molecule	force fields for molecular systems
opt	optimized versions of a few pair potentials
peri	Peridynamics model and potential
poems	coupled rigid body motion
reax	ReaxFF potential
replica	multi-replica methods
shock	methods for MD simulations of shock loading
srd	stochastic rotation dynamics (SRD)
xtc	dump atom snapshots in XTC format

- This customizability ensures that LAMMPS (in one form or another) **can be run on virtually any computer**.
- Result should be an executable located in the src/lmp_machine, where machine depends on how you compiled (for me it’s lmp_openmpi).

LAMMPS Example: How to Run

Steps:

1) Set your path to the location of the LAMMPS executable:

export PATH = /home/jason/lammps/lammps-16Apr10/src

In this "./src/" directory is the LAMMPS executable. For me: **"/src/lmp_serial"**

2) Put the example files into a directory, say:

/home/jason/lammps/LJ/0press_alat

```
root@jason-desktop:/home/jason/lammps/LJ/crystal/0press_alat# ls
in.LJAr.alat   LJ.4x4x4.in.data   tau_T10_tau_P10   tau_T1_tau_P10
```

Input Script Input Structure

3) Run the executable with:

```
root@jason-desktop:/home/jason/lammps/LJ/crystal/0press_alat# nohup lmp_serial < in.LJAr.alat &
```

This command runs LAMMPS in background and gives you back control of the command terminal. Type "top" to check if LAMMPS is running:

```
  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
31096 root      25   0  6328 1980 1416 R 99.8  0.2  29:27.65 lmp_serial
    1 root      16   0  1568  524  456 S  0.0  0.1   0:04.87 init
    2 root      34  19     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/0
    3 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 watchdog/0
    4 root      10  -5     0    0    0 S  0.0  0.0   0:00.75 events/0
```

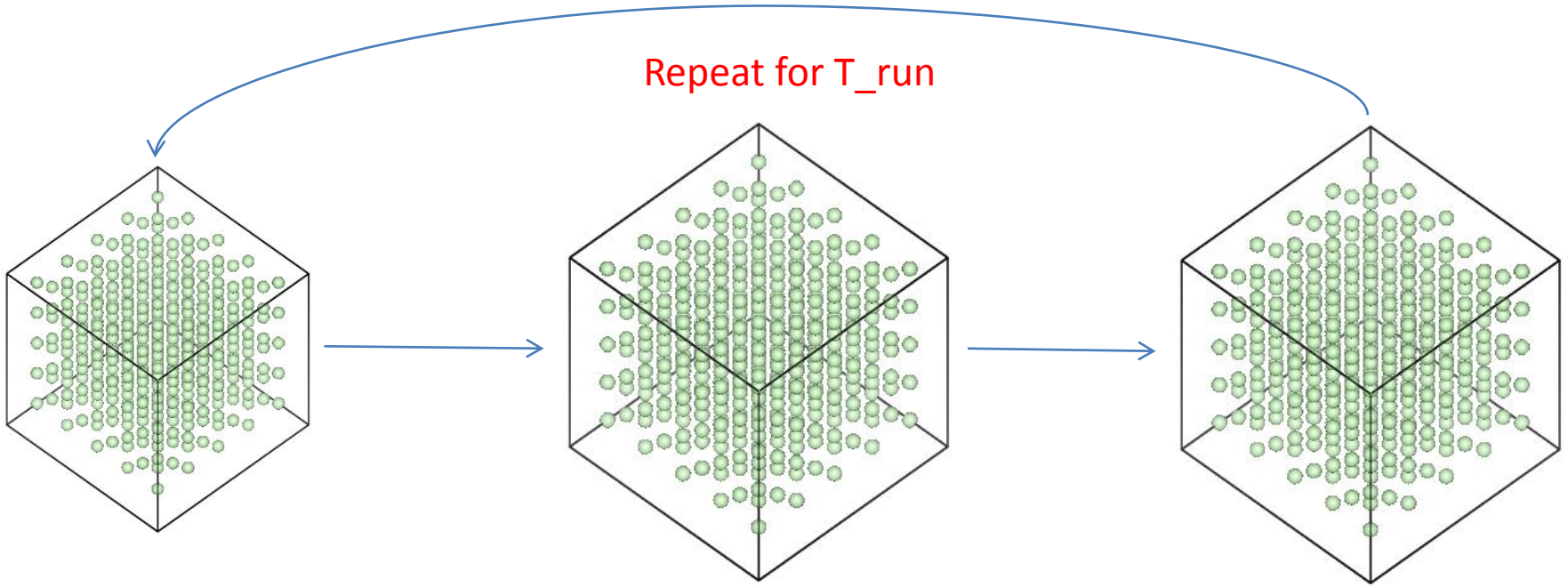
Mechanical Engineering

LAMMPS Example Code

Input file (has MD information): in.LJAr.alat Input Structure: LJ.4x4x4.in.data

What happens:

1) Measures 0 pressure lattice constant (alat) for 4x4x4 supercell of FCC Ar (256 atoms) using NPT ensemble.
2) Measure constant volume specific heat using formula: $\langle T \rangle = \left[\frac{\langle (E - \langle E \rangle)^2 \rangle}{3(N-1)k_B c_v} \right]^{1/2}$
3) Do 1) and 2) for T_run = 10...80K



NPT heat to allow cell expansion to 0 Press at T_run.

NPT data collection to measure <alat>

NVT to measure cv

Mechanical Engineering

LAMMPS Example I

```
#-------------------------------------------------------------------------------------------------
###################################MEASURE 0 PRESS ALAT#############################################
#-------------------------------------------------------------------------------------------------
clear
#------------READ STRUCTURE---------------------------------------------------------------------
units           lj                  ##Use LJ units sigma_Ar, eps_Ar, tau_Ar
atom_style      atomic                  ##Specifies how many columns in data in
read_data       LJ.4x4x4.in.data            ##Specifies input data file with "atomic"
#read_restart   ljNVE.restart.1000000
mass            1  1.0                  ##Specifies mass again, not really needed
group           Ar type = 1             ##Gives a label to atome type=1 "Ar"
#------------LJ POTENTIALS----------------------------------------------------------------------
pair_style      lj/cut 2.5              ##LJ with cutoff a=2.5
pair_coeff      1 1 1.0 1.0             ##atomi atomj epsilon sigma
pair_modify          shift yes              ##Shift PE at cutoff to 0.0

##USED TO SPECIFY MULTIPLE SPECIES
##pair_style        lj/cut 2.5
##pair_coeff        1 2 1.0 1.0
##pair_modify        shift yes

##pair_style        lj/cut 2.5
##pair_coeff        2 2 1.0 1.0
##pair_modify        shift yes

#pair_modify    tail no             ##This is default so not needed
#------------LJ PARAMETERS----------------------------------------------------------------------
variable        kB      equal 1.3806504e-23         # [J/K] Boltzmann
variable    sigma_Ar    equal 3.4e-10       # m
variable    eps_Ar      equal 1.67e-21      # J
variable    mass_Ar equal 6.63e-26      # kg
variable    tau_Ar      equal 2.1423e-12        # s
#------------THERMO PARAMETERS------------------------------------------------------------------
variable        T_melt  equal 300*(${kB}/${eps_Ar})
variable    T_OK        equal 0.001
#variable   T_run       equal 15*(${kB}/${eps_Ar})
#variable       V       equal vol
variable    dt      equal 0.002
variable    run_length  equal 500000
variable    Tau_T       equal 1.0
variable    Tau_P       equal 10.0
variable    seed        equal 99999
#------------SAMPLE PARAMETERS------------------------------------------------------------------
variable        p       equal 100           # correlation length
variable        s       equal 10            # sample interval
variable        d       equal $p*$s             # dump interval
```

in.LJAr.alat

What happens:

1) Input structure
2) Interaction potential
3) Simulation variables

Continued…

Mechanical Engineering

LAMMPS Example 2

```
#START LOOP 1

    label loop1
    variable a loop 8
    variable T_run index 0.082670659 0.165341317 0.248011976 0.330682635 0.413353293 0.496023952 &
    0.578694611 0.661365269

    log     log_$a.lammps

    #------------START NPT HEAT ------------------------------------------------------------------

        reset_timestep      0

            ##IF FIRST T_run CREATE INITIAL VEL

                ##if "${a}<2" then "velocity all create ${T_run} ${seed} rot yes dist gaussian"
                velocity all create ${T_run} ${seed} rot yes dist gaussian

    ##START
            ##RUN A PREPARATION NPT
                ##SET ENSEMBLE TO NPT

                fix             1 all npt temp ${T_run} ${T_run} ${Tau_T} iso press 0.0 ${Tau_P}
                thermo_style    custom step temp press vol etotal pe ke
                thermo      1000

                ##OPTIONAL DUMP POSITIONS

                ##dump          xyz all xyz ${run_length} npt_heat_$a.dump

                run                 50000
              #run      1000
                unfix           1
    ##STOP
```

What happens:

1) Run an NPT to initialize

Continued…

Mechanical Engineering

LAMMPS Example 3

```
#----------CALCULATE AVG CELL SIZE----------------------------------------------------------

##START
        ##RUN NPT FOR CELL SIZE
            fix             1 all npt temp ${T_run} ${T_run} ${Tau_T} iso press 0.0 ${Tau_P}

            ##SET myLx EQUAL TO THE X_DIM BOX SIZE

            variable        myLx equal xhi-xlo

            ##CREATE A FIX WHICH AVERAGES THE X-DIM BOX SIZE OVER TIME

            fix             myLx all ave/time ${p} ${s} ${d} v_myLx file Lx.profile$a

            ##SET ANOTHER VARIABLE myLx2 EQUAL TO THE TIME AVERAGED X-BOX DIM

            variable        myLx2 equal f_myLx

            ##DO FOR Y AND Z DIRECTIONS AS WELL

            variable        myLy equal yhi-ylo
            fix             myLy all ave/time ${p} ${s} ${d} v_myLy file Ly.profile$a
            variable        myLy2 equal f_myLy
            variable        myLz equal zhi-zlo
            fix             myLz all ave/time ${p} ${s} ${d} v_myLz file Lz.profile$a
            variable        myLz2 equal f_myLz

        ##REPORT THERMO DATA EVERY thermo NUMBER TIME STEPS

            thermo_style    custom step temp press vol etotal pe ke #v_myLx2 v_myLy2 v_myLz2
            thermo      1000

        ##RUN THE ENSEMBLE FOR THIS MANY TIME STEPS

            run         100000
          #run          5000

        ##UNLOCK THE DUMP AND FIX COMMANDS SO THEY CAN BE USED AGAIN LATER

            unfix           1
##STOP
```

What happens:

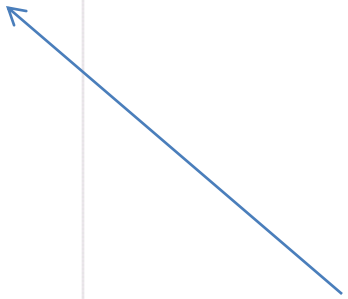
1) Run in NPT
2) Time average Lx, Ly, Lz

Continued…

Mechanical Engineering

LAMMPS Example 4

```
##START
        ##THE FINAL TEIM AVG myL(x,y,z)2 USED TO RELAX SIM BOX TO ABOVE AVG LATTICE CONTANTS

            fix             deform all deform 1 x final 0.0 $(myLx2) y final 0.0 $(myLy2) z final 0.0 $(myLz2) units box

        ##RELAX THE SIM BOX OVER THIS MANY TIME STEPS

            run         10000
            #run            1000

        ##UNLOCK FIXED AND COMPUTES TO USE LATER

            unfix       deform
            unfix       myLx
            unfix               myLy
            unfix               myLz
##STOP

#------------STOP NPT HEAT ------------------------------------------------------------------
```



What happens:

1) Sim. Box dimensions are changed to <Lx>, <Ly>, <Lz>

Continued…

Mechanical Engineering

LAMMPS Example 5

```
#------------ START NVT equil ----------------------------------------------------------

##START
        ##RUN A PREPARATION NVT
            ##SET ENSEMBLE TO NVT
            fix             1 all nvt temp ${T_run} ${T_run} ${Tau_T}
            thermo_style    custom step temp press vol etotal pe ke
            thermo      1000
            run                 50000
          #run          1000
            unfix       1
##STOP

##START
        ##COLLET NVT DATA FOR Cv
            fix             1 all nvt temp ${T_run} ${T_run} ${Tau_T}
            thermo_style    custom step temp press vol etotal pe ke
            thermo      1000
            run                 100000
          #run               5000
            unfix               1
##STOP

next T_run
next a
jump in.LJAr.alat loop1

#END LOOP 1
```

What happens:

1) Short NVT run to initialize.
2) Data collection NVT run to measure Cv
3) Loop ends for given T_run, starts over.

END

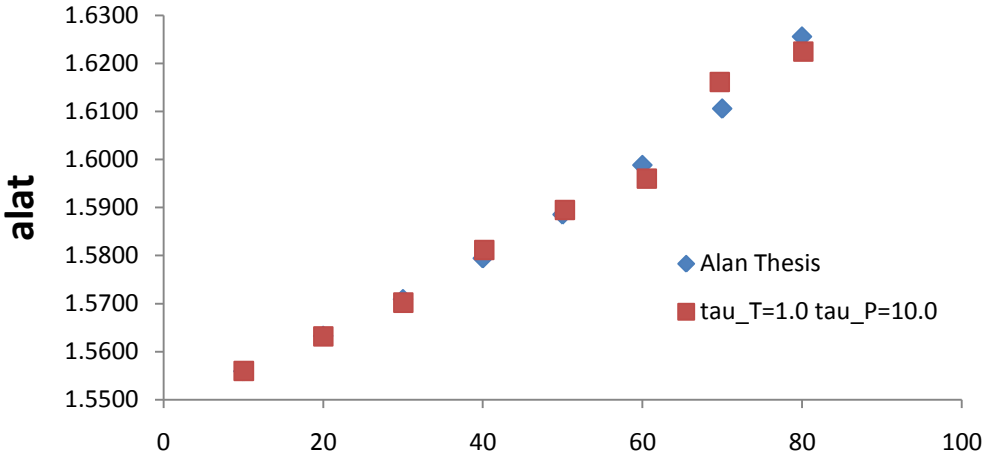
Mechanical Engineering

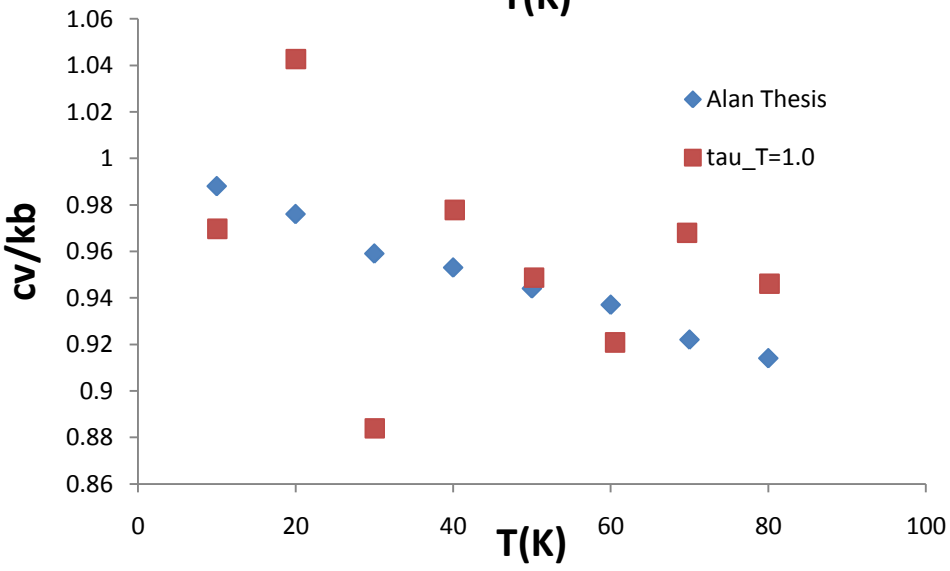
LAMMPS Example Results

Explanation for differences:

1) Thermo sampling rate (more data to average cv).
2) Average over different initial conditions (how could you do this?).
3) Vary tau_T to see if this has an effect.
4) Increase the equilibration times between NVT and NPT data collection segments.

You can easily automate all of these (and much more).





Mechanical Engineering

LAMMPS Performance

- Using the Lecture Example code, I get the following performance:

LAMMPS:

Ubuntu 9.0, 2.0 GHz CPU, 1 GB RAM, 256 atoms, NPT, neighbor listing:

Rate = 1000 steps/s

Larger systems will scale N*rate because of neighbor listing. Can be even faster using parallel capabilities.

My LJ Code:

Ubuntu 9.0, 2.0 GHz CPU, 1 GB RAM, 256 atoms, NPT, NO neighbor listing:

Rate = 70 steps/s

Larger systems will scale as ~N^2 since no neighbor listing.

LAMMPS Modifications

- This section describes how to add/modify the LAMMPS source code:

http://lammps.sandia.gov/doc/Section_modify.html

- LAMMPS is designed on a modular fashion using C++.
  http://www.cplusplus.com/doc/tutorial/

- Hi-level structure uses classes, functions are class methods and use simple array, vectors, numbers. http://www.cplusplus.com/reference/stl/vector/vector/
- Creating a new class requires new.cpp and new.h. Best way to understand is to look at existing classes (like compute_temp.cpp and compute_temp.h).
  http://www.cplusplus.com/doc/tutorial/classes/

- If worthy, new features are encouraged to be submitted to LAMMPS developers.

LAMMPS Matlab Post-Processing

- There are many ways to post-process the LAMMPS output:

http://lammps.sandia.gov/doc/Section_tools.html

- Ability to read *.dump files one at a time, or an entire run full.
- Read RDF files
- Read *.log files to easily plot thermo data