Lecture #2
Ins and Outs of LAMMPS input scripts

Steve Plimpton
Sandia National Labs
sjplimp@sandia.gov

7th LAMMPS Workshop Tutorial
Virtual meeting – August 2021

Goals for this lecture

Teach you how to ...

- Read and understand an existing input script
- Edit an existing input script
- Write a new input script
- Debug an input script

What is a LAMMPS input script

- Text file containing a sequence of LAMMPS commands
- LAMMPS reads file, executes commands one at a time
    - NOT: run one simulation after entire file is read
    - RATHER: run a simulation whenever run command appears
- One script can run one or many LAMMPS simulations
- Some commands read other flavors of LAMMPS input files
    - data files: read_data data.micelle
    - restart files: read_restart surface.restart.100000
    - molecule files: molecule ID co2.txt h2o.txt
- When last command in file completes, LAMMPS exits
    - earlier commands can also trigger exit

Input script is parsed into individual commands

See Section 5.2 of User Guide for full details

1. Blank lines are skipped
2. Comments are removed: start with # character
3. Lines ending with & character are concatenated
4. Now have a single line $\Rightarrow$ single command

Input script is parsed into individual commands

See Section 5.2 of User Guide for full details

1. Blank lines are skipped
2. Comments are removed: start with # character
3. Lines ending with & character are concatenated
4. Now have a single line ⇒ single command
5. Single line is split into words by white space
6. Quotes allow one word to contain spaces
    - two words: print “Reached end of equilibration”
7. Within each word, variable names replaced with value
    - variable f string mydata.micelle
    - read_data $f
    - variable temp equal 273.0
    - fix ID all nvt temp ${temp} ${temp} 0.01

Input script is parsed into individual commands

See Section 5.2 of User Guide for full details

1. Blank lines are skipped
2. Comments are removed: start with # character
3. Lines ending with & character are concatenated
4. Now have a single line ⇒ single command
5. Single line is split into words by white space
6. Quotes allow one word to contain spaces
   - two words: print "Reached end of equilibration"
7. Within each word, variable names replaced with value
   - variable f string mydata.micelle
   - read_data $f
   - variable temp equal 273.0
   - fix ID all nvt temp ${temp} ${temp} 0.01
8. First word = command name, all others are arguments

Every command has its own doc page

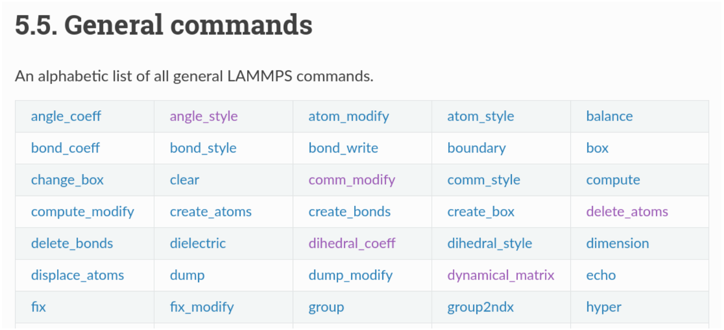
IMPORTANT:

The only way to edit/compose your own input scripts and
learn how to use LAMMPS well, is to read those doc pages

Every command has its own doc page

IMPORTANT:
The only way to edit/compose your own input scripts and learn how to use LAMMPS well, is to read those doc pages

Bookmark Section 5.5 of User Guide:



5.5. General commands

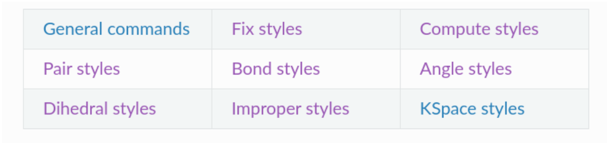
An alphabetic list of all general LAMMPS commands.

angle_coeff	angle_style	atom_modify	atom_style	balance
bond_coeff	bond_style	bond_write	boundary	box
change_box	clear	comm_modify	comm_style	compute
compute_modify	create_atoms	create_bonds	create_box	delete_atoms
delete_bonds	dielectric	dihedral_coeff	dihedral_style	dimension
displace_atoms	dump	dump_modify	dynamical_matrix	echo
fix	fix_modify	group	group2ndx	hyper

and many more (∼110 commands in current version)

Some style commands are actually many commands

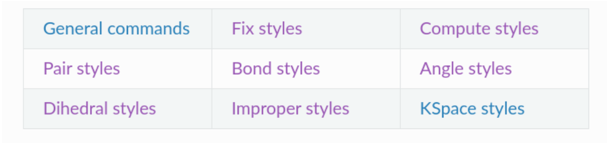
Selections at very top are style commands



General commands	Fix styles	Compute styles
Pair styles	Bond styles	Angle styles
Dihedral styles	Improper styles	KSpace styles

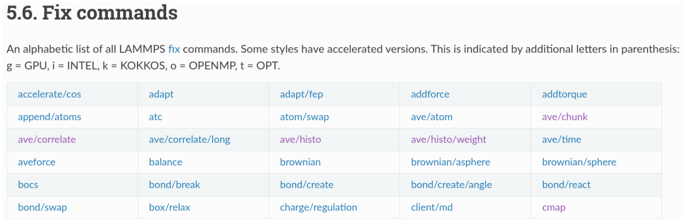
Some style commands are actually many commands

Selections at very top are style commands



General commands	Fix styles	Compute styles
Pair styles	Bond styles	Angle styles
Dihedral styles	Improper styles	KSpace styles

Expands to ∼235 entries:



5.6. Fix commands

An alphabetic list of all LAMMPS fix commands. Some styles have accelerated versions. This is indicated by additional letters in parenthesis: g = GPU, i = INTEL, k = KOKKOS, o = OPENMP, t = OPT.

accelerate/cos	adapt	adapt/fep	addforce	addtorque
append/atoms	atc	atom/swap	ave/atom	ave/chunk
ave/correlate	ave/correlate/long	ave/histo	ave/histo/weight	ave/time
aveforce	balance	brownian	brownian/asphere	brownian/sphere
bocs	bond/break	bond/create	bond/create/angle	bond/react
bond/swap	box/relax	charge/regulation	client/md	cmap

A command like: fix ID all aveforce ...

may be called a fix command or fix aveforce command

Structure of a typical input script

1. Global settings
    - units, dimension, atom_style, boundary commands
    - all have default values
2. Create simulation box and atoms
    - region, create_box, create_atoms, lattice commands
    - read_data or read_restart commands
3. Define groups of atoms
    - one atom can be assigned to multiple groups
4. Set atom attributes if needed: velocity, mass
5. Pair_style command for atom interactions
6. Fix commands for time integration and constraints
7. Compute commands for diagnostics
8. Output commands: thermo, dump, restart
9. Action command: run or minimize
10. Rinse and repeat as needed

Friction input script – section 1

Global settings:

```
dimension 2
boundary p s p
atom_style atomic
neighbor 0.3 bin
neigh_modify delay 5
timestep 0.0025
```

Friction input script – section 2

Create box and atoms:

```
lattice hex 0.9
region box block 0 50 0 22 -0.25 0.25
create_box 4 box

region lo-fixed block INF INF INF 1.1 INF INF
region lo-slab block INF INF INF 7 INF INF
region above-lo block INF INF INF 7 INF INF side out
region hi-fixed block INF INF 20.9 INF INF INF
region hi-slab block INF INF 15 INF INF INF
region below-hi block INF INF 15 INF INF INF side out

create_atoms 1 region lo-slab
create_atoms 1 region hi-slab
```

Friction input script – section 3

Define groups:

```
group lo region lo-slab
group lo type 2
group hi region hi-slab
group hi type 3
group lo-fixed region lo-fixed
group hi-fixed region hi-fixed
group boundary union lo-fixed hi-fixed
group mobile subtract all boundary
```

Friction input script – section 4

Set atom attributes:

```
set group lo-fixed type 4
set group hi-fixed type 4

mass * 1.0
velocity mobile create 0.1 482748 temp ydim
velocity hi set 1.0 0.0 0.0 sum yes
```

Friction input script – section 5

Pair_style:

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

Friction input script – section 6

Fixes:

```
fix 1 all nve
fix 2 boundary setforce 0.0 0.0 0.0
fix 3 mobile temp/rescale 200 0.1 0.1 0.02 1.0
fix_modify 3 temp ydim
fix 4 all enforce2d
```

Friction input script – section 7

Single compute:

```
compute ydim mobile temp/partial 0 1 0
```

Friction input script – section 8

Output:

```
thermo 1000
thermo_modify temp ydim

dump 2 all image 500 image.*.jpg type type &
zoom 1.6 adiam 1.5
dump_modify 2 pad 5

dump 3 all movie 500 movie.mpg type type &
zoom 1.6 adiam 1.5
dump_modify 3 pad 5
```

Friction input script – section 9

Single action:

```
run 20000
```

Log file output = log.lammps

See Section 4.3 of User Guide for full details

- Similar to screen output but some additional info
- Contains warning and error messages
- Echoes every command including variable substitutions
- Many input script commands produce useful output
  - group command prints # of atoms in group
  - delete_atoms command prints # of atoms deleted

Log file output = log.lammps

See Section 4.3 of User Guide for full details

- Similar to screen output but some additional info
- Contains warning and error messages
- Echoes every command including variable substitutions
- Many input script commands produce useful output
  - group command prints # of atoms in group
  - delete_atoms command prints # of atoms deleted
- Pre-run info: neighbor lists, memory usage
- Columns of numeric thermodynamic output every N steps
  - state of simulation at that timestep (not just thermo)
  - thermo_style command defines what info is output
  - useful to eyeball or plot
- Post-run info:
  - CPU timing and breakdown (pair, neighbor, comm, etc)
  - per-processor stats on atom/neighbor counts & timings
  - useful to eyeball for performance issues

Dump file output

- Dump file = one snapshot of per-atom info every N steps
- Input script can specify multiple dump commands
- Choose which atoms and what info to output
- Many fix, compute, variable commands $\Rightarrow$ per-atom values
- Can also output per-pair or per-bond info
- Useful for viz and post-processing analyses

Dump file output

- Dump file = one snapshot of per-atom info every N steps
- Input script can specify multiple dump commands
- Choose which atoms and what info to output
- Many fix, compute, variable commands $\Rightarrow$ per-atom values
- Can also output per-pair or per-bond info
- Useful for viz and post-processing analyses

Fix time-averaging commands also produce output files

- fix ave/time - scalar or vector quantities
- fix ave/histo - histograms of per-atom data
- fix ave/correlate - correlation coeffs of scalars or vectors

(1) Tips for writing and debugging an input script

- Same as writing a computer program
  - albeit in a simple input-script language
- Start as simple as possible
- Add complexity to your script one command at a time
- Check thermo output (plot) and/or viz at every stage
- Two kinds of errors: compile (syntax) and run-time
- LAMMPS flags syntax errors, tells you which command
  - common: LAMMPS not built with needed package
  - if error not obvious, read the command doc page
  - NOTE: webpage manual is for current patch version
- Pay attention to screen/logfile WARNING messages

(2) Tips for writing and debugging an input script

- Debug tools for run-time errors:
  - read pre-run portion of log file to insure all-is-well
  - add variables and print commands to examine values
  - run with thermo output every timestep
  - run in serial and parallel
  - plot thermo output, viz snapshots to verify all-is-well
- Four flavors of run-time errors, from easy to hard
  1. Simulation triggers an error messsage:
     - common: lost or out-of-range atoms
     - cause: overlapping atoms or bad model params
     - will show how to find additional info later
  2. Simulation thermodynamics blow up
     - common: bad model or time integration params
  3. Simulation crashes with no error message or blow-up
     - if thermo output looks good, report it
  4. See next slide ...

Hardest run-time error to debug

Simulation runs, answer is wrong

- How do you know what is right versus what is wrong ?
- Could be normal statistical variation
- Could be LAMMPS did what you asked, your model is wrong
- Could be a bug or problem with LAMMPS
- Hard to deduce without some MD expertise

Hardest run-time error to debug

Simulation runs, answer is wrong

- How do you know what is right versus what is wrong ?
- Could be normal statistical variation
- Could be LAMMPS did what you asked, your model is wrong
- Could be a bug or problem with LAMMPS
- Hard to deduce without some MD expertise

When all else fails ...

- Ask a local LAMMPS or MD expert (your advisor?)
- Post a message to mail list or forum
- See website Mail list or MatSci forum for details
- Read Mail list guidelines link (top 10 list)