# Running LAMMPS with Modulefiles

**Notes by Shane Flynn,** sflynn@caltech.edu

I have worked with Shyam Saladi (saladi@caltech.edu) to develop an efficient method to run the necessary software to perform LAMMPS calculations. LAMMPS is an open-source code that is constantly changing; it would be inconvenient for each user/student in the group to need to compile and configure (*i.e.*, set environment variables) the software whenever an update occurs.

When loading software, many parts of a user's environment variables must be modified. When setting up software, some commonly set environment variables include the following: PATH, LD_LIBRARY_PATH, MANPATH. For example, the PATH specifies where the current shell (*e.g.* bash, csh, zsh) looks for executable files.

One solution to this problem is to use modulefiles. Environment Modules is a software package that dynamically modify a user's environment variables. With a single command, it modifies/unmodifies your environment to easily run the software you are hoping to use. Modules is a organized way to manage different versions of a single piece of software and software that has conflicting dependencies.

In the 2pt homework assignment, we will use LAMMPS to run molecular dynamics calculations. To do these calculations, we will copy Shyam's modulefiles configuration to bring software he has installed/configured into view. If a new version is installed, Shyam can update his module files and then everyone with his configuration will also have access to the updated software. Log into ion or atom and type the following:

```
% module load use.own
% cp ~ch121sms/privatemodules/saladimodules ~/privatemodules
```

These two lines will make a directory called privatemodules on your computer (you only need to do this once and then never again!). Now when we want to submit a LAMMPS calculation to the cluster, we need to load this module to bring the software into view. Inside of your submission script, the following lines need to be added to the bottom:

```
% module load use.own
% module load saladimodules
% module load openmpi-x86_64
% module load lammps

% mpirun -np 4 lmp -in in.waterbox
```

The first 4 lines load various modules Shyam has constructed for maintaining software: use.own tells the system to look in our privatemodules directory (the one created above). Next we load the saladimodules to bring Shyam's modules into view. OpenMPI is a software package that provides parallel computing capability. This is necessary to run LAMMPS across many/all processors on a compute node. Therefore, we load this and then, finally, LAMMPS. The last line starts a samplecalculation; to be clear, mpirun is from the openmpi-x86_64 module, lmp is from the lammps module). In this case we tell mpi to use 4 processors to run LAMMPS (lmp) with an in file called in.waterbox

## Module Commands

Now that we have Shyam's modules configured, we can use some commands to explore the set-up. While logged into ion or atom, at the console, type the following:

```
% module avail
```

As you guessed, this command shows the modules that are available to you. It will list every module available, including each version of the software.

Remember, all module files do is temporarily change environment variables. In particular, by changing the PATH variable, executables for software you want comes into view. Therefore, you make your own modulefile to point to a location where software is updated and maintained. Modules are written in a language known as tcl, the files themselves are easy to understand. You can painlessly write your own module for new software you'd like to set-up for yourself.

To load a module at the console, simply type

```
% module load 'name'/'version'
```

To see which modules you currently have loaded, you can use

```
% module list
```

After loading the specified modules above, `module list` should return 5 different modules (`rocks-openmpi`, `use.own`, `saladimodules` `openmpi-x86_64`, and `lammps/2015May15`).

With the specified modules loaded, we are now ready to run LAMMPS molecular dynamics calculations!