

# MODULE 2: MOLECULAR DYNAMICS

Practice: LAMMPS

# **I. What is LAMMPS?**

# LAMMPS

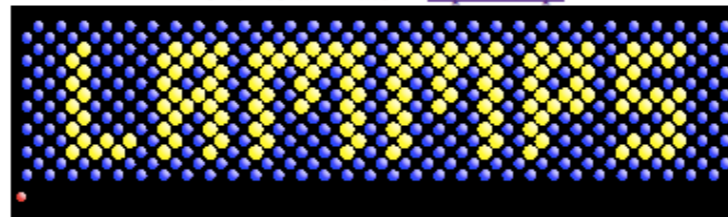
<http://lammps.sandia.gov>

Large-scale **A**tomic/**M**olecular **M**assively **P**arallel **S**imulator

## LAMMPS Molecular Dynamics Simulator

*lamp*: a device that generates light, heat, or therapeutic radiation; something that illumines the mind or soul -- [www.dictionary.com](http://www.dictionary.com)

hover to animate -- [input script](#)



[physical analog \(start at 3:25\)](#) & [explanation](#)

Big Picture	Code	Documentation	Results	Related Tools	Context	User Support
<a href="#">Features</a>	<a href="#">Download</a>	<a href="#">Manual</a>	<a href="#">Publications</a>	<a href="#">Pre/Post Processing</a>	<a href="#">Authors</a>	<a href="#">Mail list</a>
<a href="#">Non-features</a>	<a href="#">SourceForge</a>	<a href="#">Developer Guide</a>	<a href="#">Pictures</a>	<a href="#">Pizza.py Toolkit</a>	<a href="#">History</a>	<a href="#">Workshops</a>
<a href="#">FAQ</a>	<a href="#">Latest Features &amp; Bug Fixes</a>	<a href="#">Tutorials</a>	<a href="#">Movies</a>	<a href="#">Offsite LAMMPS packages &amp; tools</a>	<a href="#">Funding</a>	<a href="#">User Scripts</a> and <a href="#">HowTos</a>
<a href="#">Wish list</a>	<a href="#">Unfixed bugs</a>	<a href="#">MD to LAMMPS glossary</a>	<a href="#">Benchmarks</a>	<a href="#">Visualization</a>	<a href="#">Open source</a>	<a href="#">Contribute to LAMMPS</a>
.	.	<a href="#">Commands</a>	<a href="#">Citing LAMMPS</a>	<a href="#">Related Modeling codes</a>	.	.

LAMMPS is a classical molecular dynamics code, and an acronym for Large-scale Atomic/Molecular Massively Parallel Simulator.

LAMMPS has potentials for solid-state materials (metals, semiconductors) and soft matter (biomolecules, polymers) and coarse-grained or mesoscopic systems. It can be used to model atoms or, more generically, as a parallel particle simulator at the atomic, meso, or continuum scale.

LAMMPS runs on single processors or in parallel using message-passing techniques and a spatial-decomposition of the simulation domain. The code is designed to be easy to modify or extend with new functionality.

LAMMPS is distributed as an [open source code](#) under the terms of the [GPL](#). The current version can be downloaded [here](#). Links are also included to older F90/F77 versions. Periodic releases are also available on [SourceForge](#).

LAMMPS is distributed by [Sandia National Laboratories](#), a US [Department of Energy](#) laboratory. The main authors of LAMMPS are listed on [this page](#) along with contact info and other contributors. Funding for LAMMPS development has come primarily from DOE (OASCR, OBER, ASCI, LDRD, Genomes-to-Life) and is [acknowledged here](#).

The LAMMPS WWW site is hosted by Sandia, which has this [Privacy and Security statement](#).

# History

- Born mid-90's in cooperation between Sandia, LLNL, Cray, Bristol Meyers Squibb, and Dupont — now developed at Sandia under DOE funding
- Current release in C++ w/ MPI
- **Open source and free under GPL**



# Installation

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

## Download LAMMPS

There are several ways to get the LAMMPS software.

You can follow the download instructions on this page to grab a tarball, and then follow the instructions in [Section Getting Started](#) of the LAMMPS manual to use "make" and build an executable for any machine.

If you have Subversion (SVN) or Git installed on your machine, you can use checkout and update commands to get the LAMMPS files once and then stay current. You then build LAMMPS as you would with the tarball. Further instructions on this for [SVN](#) and [Git](#) are below.

If you are on a Linux box, you can download a pre-built executable. For Ubuntu it is a personal package archive (PPA). For Fedora/RedHat/CentOS/openSUSE it is a binary RPM. The executable includes all LAMMPS packages that do not use additional libraries from the lib directory (e.g. MEAM, GPU, USER-CUDA, etc) and is kept up-to-date daily. Further instructions on this are below for [Ubuntu](#) and [RPMs](#)

OS X users can use the popular package manager [Homebrew](#) to install LAMMPS, the Python module, and additional files and resources (i.e. potential files, tools, etc). Further instructions on this are below for [OS X with Homebrew](#).

Windows users can download a serial or parallel executable below in the [Download section](#). Note that these versions are only updated infrequently. Instead, we recommend you download installer packages for 32-bit or 64-bit executables as described below in the [Windows installer](#) section. These versions are updated continuously, similar to the development version of LAMMPS, with new bug fixes and features.

- [Download a tarball](#)
- [SVN checkout and update](#)
- [Git checkout and update](#)
- [Pre-built Ubuntu executables](#)
- [Pre-built binary RPMs for Fedora/RedHat/CentOS/openSUSE](#)
- [Pre-built Gentoo executable](#)
- [OS X with Homebrew](#)
- [Windows installer packages for 32-bit and 64-bit](#)
- [Applying patches](#)

■ Platforms: Linux, Mac, Windows

■ Format: exe, RPM, PPA, SVN, Git, Homebrew, tarball

# Documentation

■ Excellent manual  
(<http://lammps.sandia.gov/doc/Manual.html>)

■ Introductory Tutorials and HowTos  
(<http://lammps.sandia.gov/howto.html>)

Big Picture	Code	Documentation	User Support
<a href="#">Features</a>	<a href="#">Download</a>	<a href="#">Manual</a>	<a href="#">Mail list</a>
<a href="#">Non-features</a>	<a href="#">SourceForge</a>	<a href="#">Developer Guide</a>	<a href="#">Workshops</a>
<a href="#">FAQ</a>	<a href="#">Latest Features &amp; Bug Fixes</a>	<a href="#">Tutorials</a>	<a href="#">User Scripts</a> and <a href="#">HowTos</a>
<a href="#">Wish list</a>	<a href="#">Unfixed bugs</a>	<a href="#">MD to LAMMPS glossary</a>	<a href="#">Contribute to LAMMPS</a>
.	.	<a href="#">Commands</a>	.

1. [Introduction](#)
  - 1.1 [What is LAMMPS](#)
  - 1.2 [LAMMPS features](#)
  - 1.3 [LAMMPS non-features](#)
  - 1.4 [Open source distribution](#)
  - 1.5 [Acknowledgments and citations](#)
2. [Getting started](#)
  - 2.1 [What's in the LAMMPS distribution](#)
  - 2.2 [Making LAMMPS](#)
  - 2.3 [Making LAMMPS with optional packages](#)
  - 2.4 [Building LAMMPS via the Make.py script](#)
  - 2.5 [Building LAMMPS as a library](#)
  - 2.6 [Running LAMMPS](#)
  - 2.7 [Command-line options](#)
  - 2.8 [Screen output](#)
  - 2.9 [Tips for users of previous versions](#)
3. [Commands](#)
  - 3.1 [LAMMPS input script](#)
  - 3.2 [Parsing rules](#)
  - 3.3 [Input script structure](#)
  - 3.4 [Commands listed by category](#)
  - 3.5 [Commands listed alphabetically](#)
4. [Packages](#)
  - 4.1 [Standard packages](#)
  - 4.2 [User packages](#)

■ Friendly user base and mailing list  
(<http://lammps.sandia.gov/mail.html>)

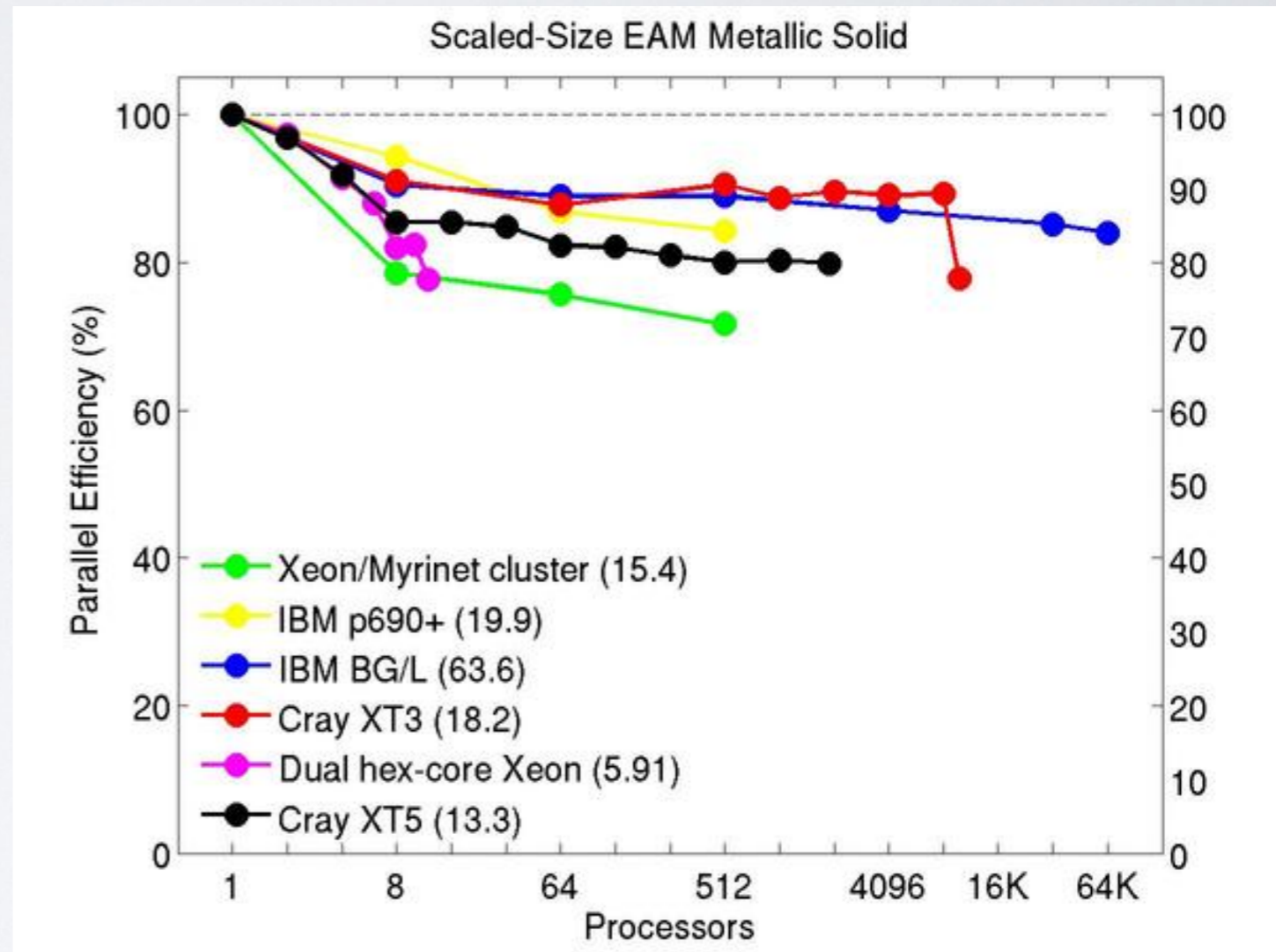
■ Excellent third-party tutorials hosted by CAVS @ MSU  
([https://icme.hpc.msstate.edu/mediawiki/index.php/LAMMPS\\_tutorials](https://icme.hpc.msstate.edu/mediawiki/index.php/LAMMPS_tutorials))

# Performance

- Extremely fast, efficient, and scalable
  - MPI parallelism
  - GPU CUDA support for most routines

## Weak scaling parallel efficiency:

- Cu metallic solid w/ EAM potential
- 32k atoms per processor (e.g., 64k proc run = 2 billion atoms)
- $P.E.(N) = \frac{t_1}{t_N}$
- Single processor timings in seconds in parentheses



## **II. Running LAMMPS**



# Usability

- Run initialization and control via **input script**

- Call from command line as `./lmp < in.comp`

- No GUI**, but some python tools available

([http://lammmps.sandia.gov/doc/Section\\_python.html](http://lammmps.sandia.gov/doc/Section_python.html))

```
Al_fcc.in
# ----- Initialize Simulation -----
units metal
dimension 3
boundary p p p
atom_style atomic

# ----- Create Atoms -----
lattice      fcc 4
region box block 0 1 0 1 0 1 units lattice
create_box  1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ----- Define Interatomic Potential -----
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ----- Define Settings -----
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ----- Dump Options -----
dump      1 all atom 1 dump.relax

# ----- Run Minimization -----
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

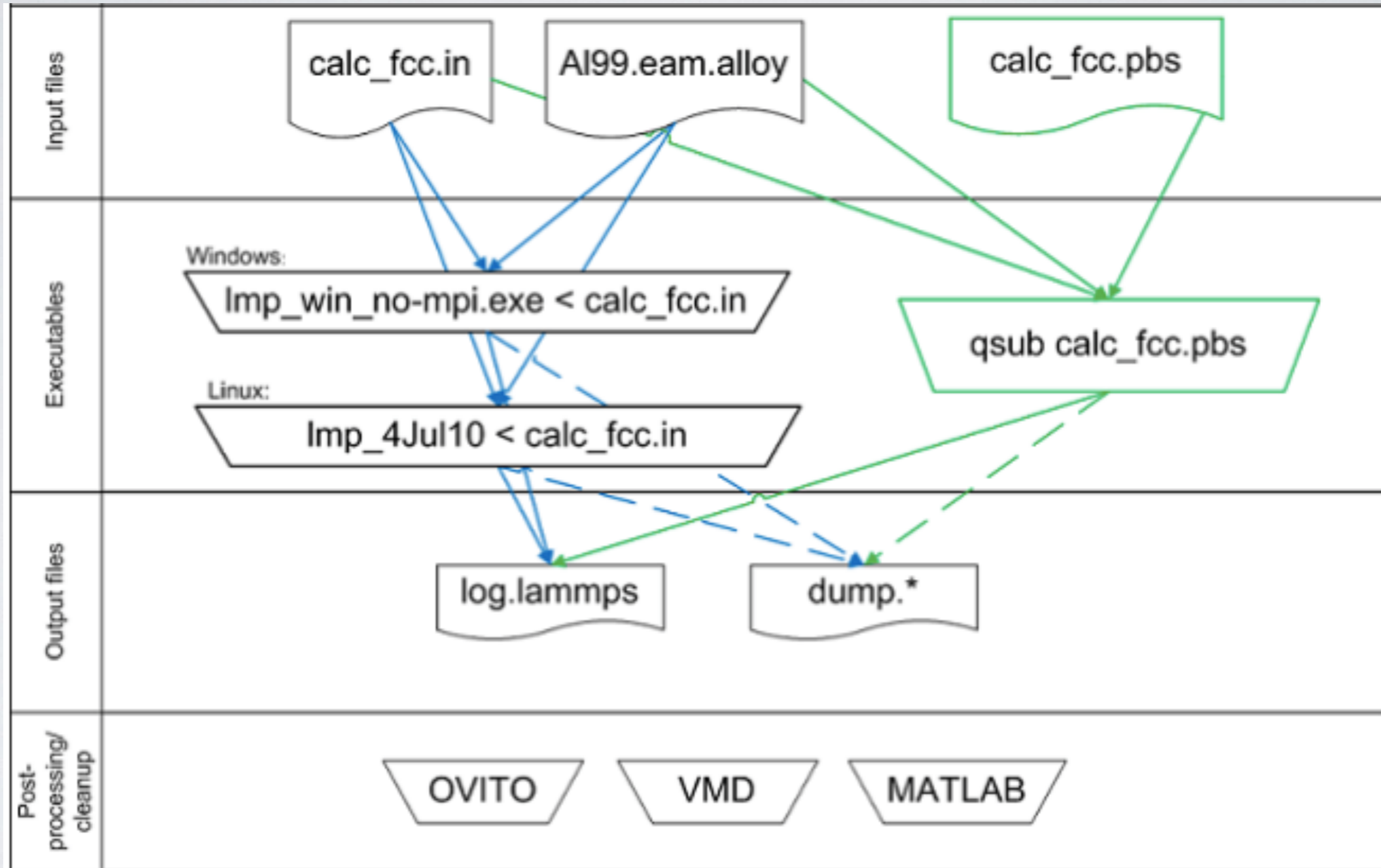
print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstroms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

# Anatomy of a LAMMPS simulation

local run

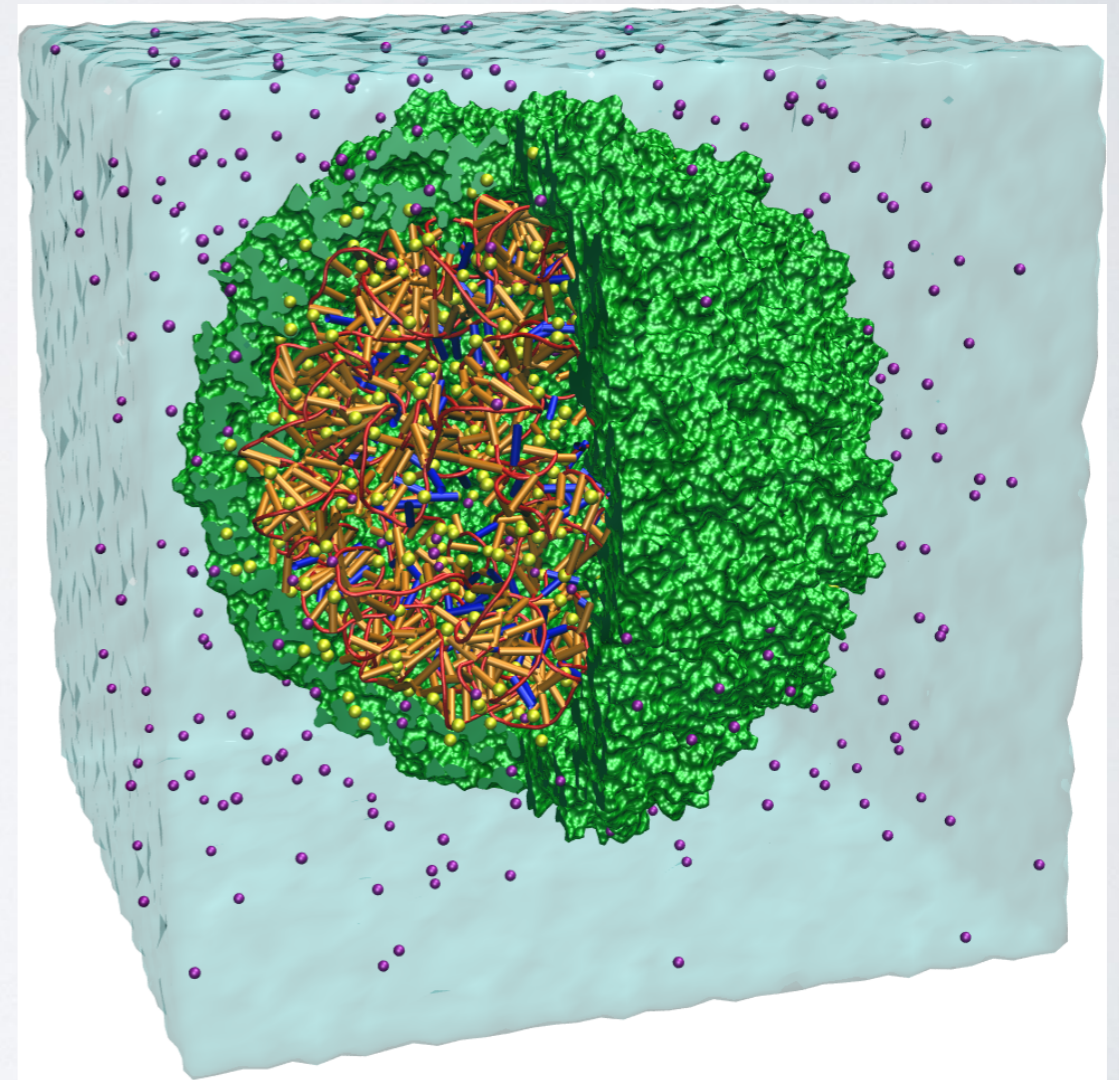
batch job



# **III. Visualizing trajectories**

# Visualization

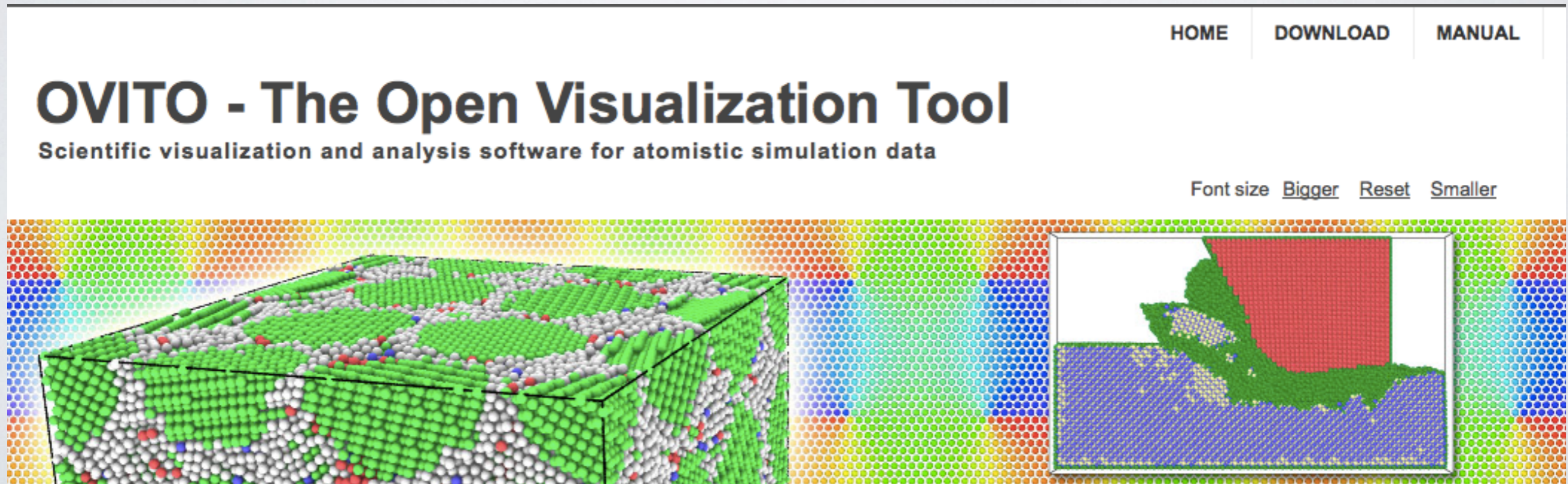
- LAMMPS contains no built-in visualization program
- Trajectory visualization is an extremely important part of molecular simulation:
  - “sanity check”
  - error diagnosis
  - global overview of structure and dynamics
  - guide and inform where to study / analyze
  - required for presentations of MD research!



Theoretical and Computational Biophysics Group  
Beckman Institute  
University of Illinois at Urbana-Champaign

# OVITO

- Many visualization programs exist (e.g., VMD, Chime, Rasmol, Chimera, PyMol)
- **OVITO** is a free, user-friendly and powerful visualization engine available for Linux, Mac and Windows that plays well with LAMMPS



<http://www.ovito.org>

# OVITO

The screenshot displays the OVITO (Open Visualization Tool) interface. The main window is titled "Ovito (Open Visualization Tool)". The interface is divided into several sections:

- Top Viewport:** Shows a top-down view of the simulation cell, which is a rectangular prism containing a dense packing of particles. The particles are colored based on a property, with a gradient from blue to red.
- Front Viewport:** Shows a front view of the simulation cell, highlighting the internal structure and the distribution of colored particles.
- Left Viewport:** Shows a left-side view of the simulation cell, providing another perspective of the particle arrangement.
- Perspective Viewport:** Shows a 3D perspective view of the simulation cell, allowing for a more comprehensive understanding of the simulation's geometry and the distribution of particles.

The right-hand panel contains the following sections:

- dump.deform\_8625.cfg [CFG]:** The current configuration file being loaded.
- Add modification...:** A list of modifications to be applied to the simulation. The "Display" section includes "Simulation cell" and "Particles". The "Modifications" section includes "Color coding".
- Color coding:** A section for configuring the color coding of the particles. It includes a "Property" dropdown set to "csym", a "Color gradient" dropdown set to "Rainbow", an "End value" of 12, and a "Start value" of 0. A color gradient legend is shown below these settings. Buttons for "Adjust range" and "Reverse range" are also present.
- Color only selected particles:** A checkbox that is currently unchecked.
- Keep selection:** A checkbox that is currently unchecked.
- Render color legend (experimental):** A checkbox that is currently unchecked.

The bottom toolbar includes a timeline with a scale from 0 to 60, a "70 / 80" indicator, and various navigation and visualization controls such as "Home", "Previous", "Next", "End", "Zoom", "Rotate", and "Reset".