

# Campinas Advanced School of Thermodynamics

## Workshop on Molecular Simulations

### Session 2 –MD of ethanol with MoSDeF

Edward Maginn and Eliseo Marin

Department of Chemical and Biomolecular Engineering

University of Notre Dame, USA

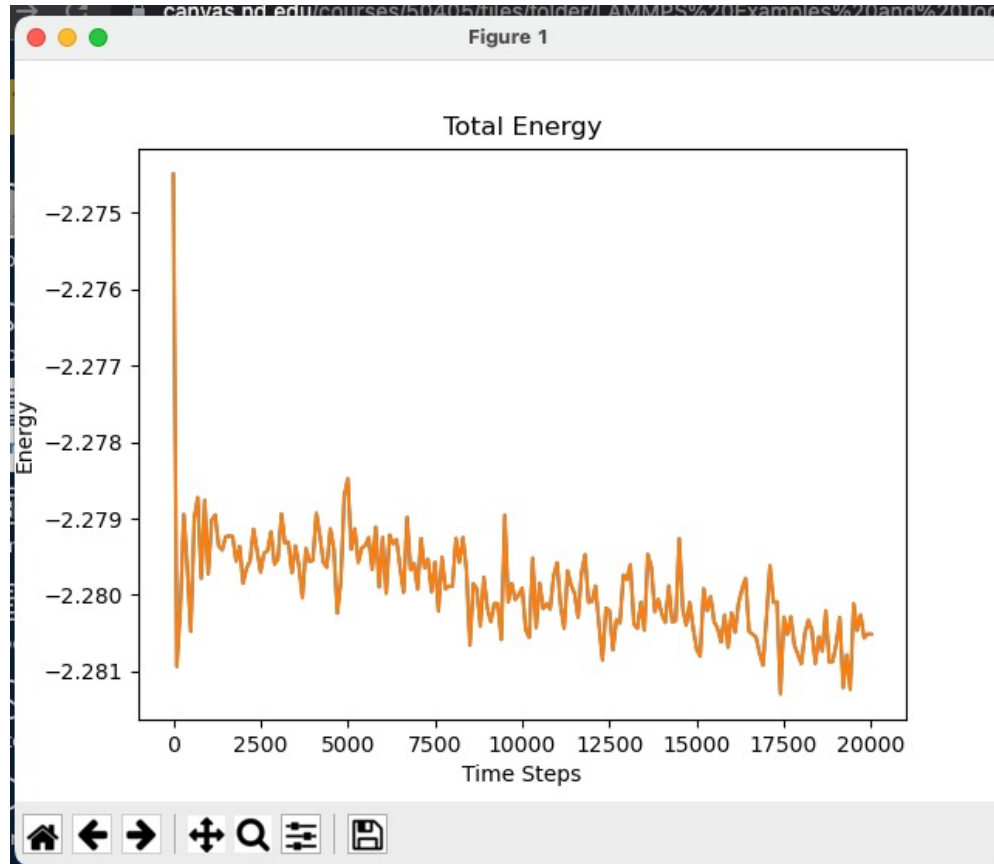
ed@nd.edu

All materials available on  
MaginnGroup GitHub site

28 October, 2022



# Computing properties

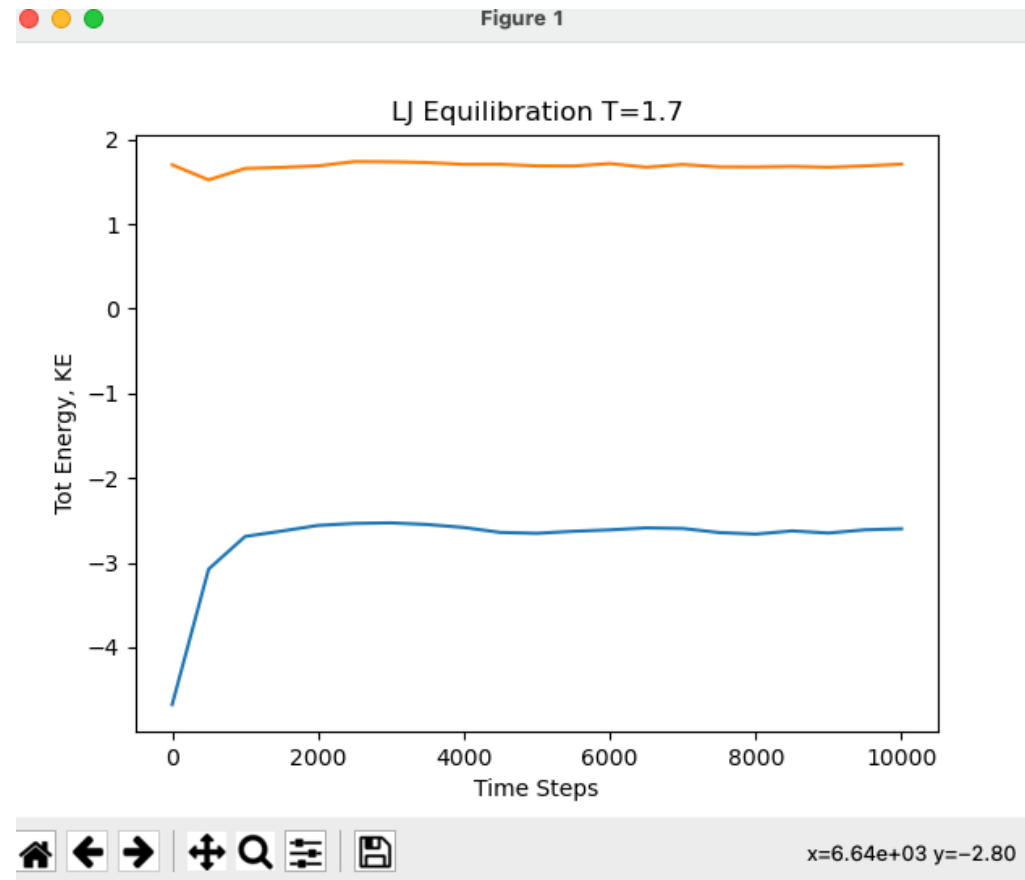


System is clearly “not equilibrated”

## Procedure:

1. Run an “equilibration phase” where system relaxes to equilibrium. Throw this out.
2. Restart from the end of equilibration run, compute properties from this.

# Longer run



Average Energy -2.730  
Average Temperature 1.686

Incorrect! Don't include part that  
is not equilibrated!

```
# To begin simulation from a restart file, uncomment the line below (adjusting
# the file name, if necessary) and comment the section corresponding to
# information now provided by the restart file.
```

```
#read_restart ${NAME}.restart.*
```

```
# #####
# start of information provided by restart file
# (If reading data from restart file, comment this section.)
```

```
# configure and initialize system
units lj
atom_style atomic
pair_style lj/cut 2.5 # style args(=cutoff)
boundary p p p # x y z (p = periodic boundary conditions)
```

```
lattice fcc 0.8442 # style scale(=reduced_density)
region MYBOX block 0 10 0 10 0 10 # ID style args(=xlo,xhi,ylo,yhi,zlo,zhi)
create_box 1 MYBOX # number_of_atom_types region_ID
```

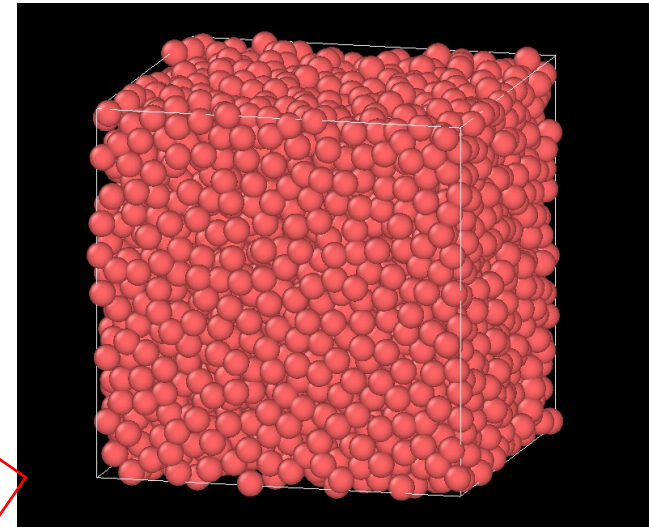
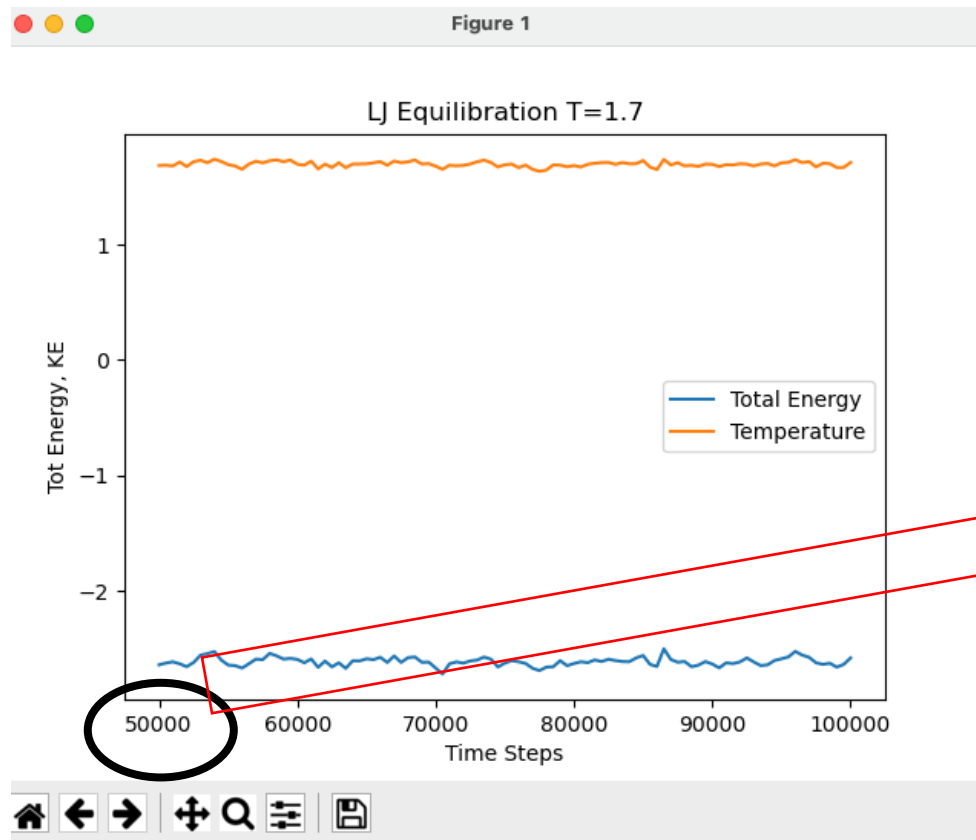
```
create_atoms 1 box # atom_type style args
```

```
mass 1 1.0 # atom_type mass
velocity all create 3.0 87287 # group_ID style args(=temp,seed)
```

```
# set forcefield parameters
pair_coeff 1 1 1.0 1.0 2.5 # atom_i atom_j args(=epsilon,sigma,cutoff)
```

```
# end of information provided by restart file
# #####
```

# Restart run

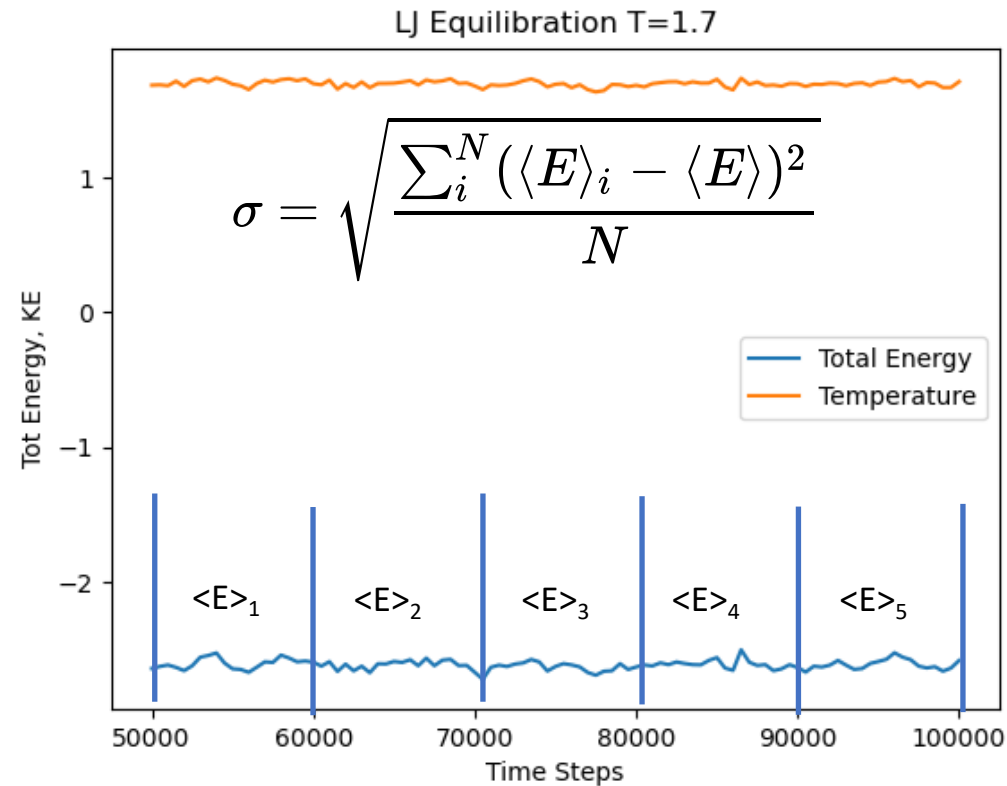


Average Energy -2.6109  
Average Temperature 1.694

# Proper averages and uncertainties

- Always make sure you take averages over equilibrated trajectories
  - Initial conditions are "irrelevant"
  - Like to start simulations "close" to equilibrium
  - Use restart files from other simulations close to desired statepoint
- How long should you average?
  - Longer than longest "correlation time" of system
  - Make sure running average is not changing with time
  - Rule of thumb: make sure molecules move several molecular diameters during simulation
- Uncertainties
  - Standard deviation from multiple independent simulations
  - "Block average" standard deviation

# Block averages



- Blocks assumed to be independent (longer than correlation time)
- Change block size and check for convergence

# Mixtures

Typically relate unlike atom parameters to a combination of like parameters (combination or mixing rule):

$$\epsilon_{ab} = (\epsilon_{aa} \epsilon_{bb})^{1/2}$$

“Lorentz-Berthelot”

$$\sigma_{ab} = (\sigma_{aa} + \sigma_{bb})/2$$

Several other possibilities (purely geometric, “size weighted”, “Kong”, etc.)

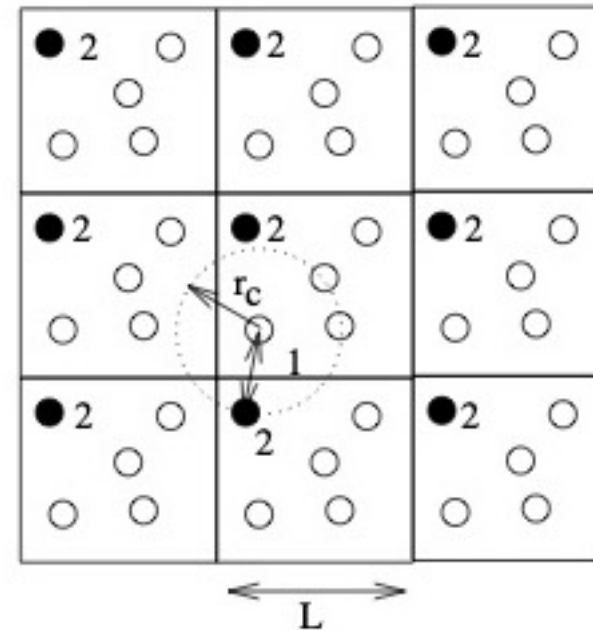
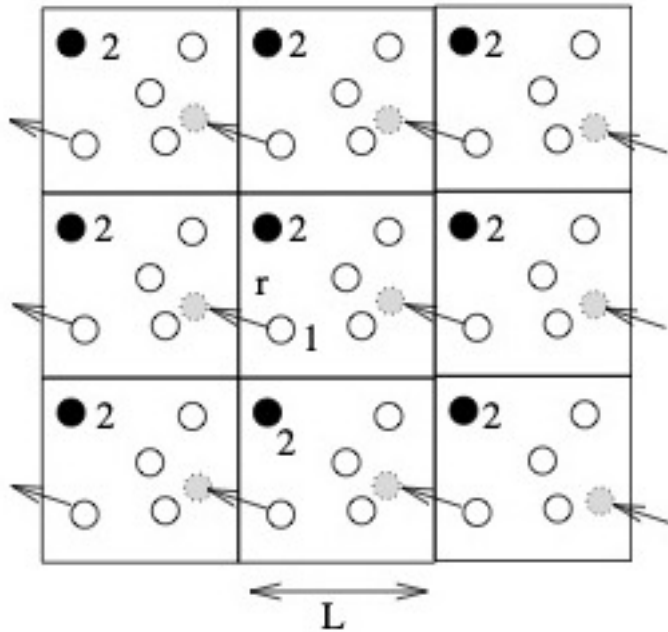
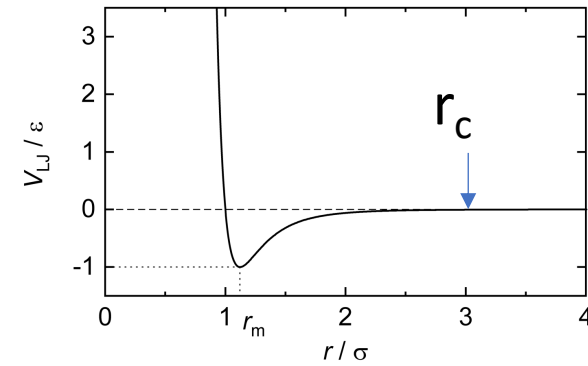
#LAMMPS mixing rules

`mix geometric`

`mix arithmetic`     (*this is Lorentz-Berthelot*)



# Potential Truncation



Cut off potential at no more than  $\frac{1}{2}$  box edge length (LJ only – charges are handled differently)

# Account for energy beyond $r_c$

Long range (“tail”) corrections

$$E_{tot} = E_{cut} + E_{LRC} = E_{cut} + 2\pi N\rho \int_{r_{cut}}^{\infty} r^2 \mathcal{V}(r) dr$$

For the LJ potential, this is analytic

$$E_{LRC} = 8\pi\rho N\epsilon \left[ \frac{\sigma^{12}}{9r_c^9} - \frac{\sigma^6}{3r_c^3} \right]$$

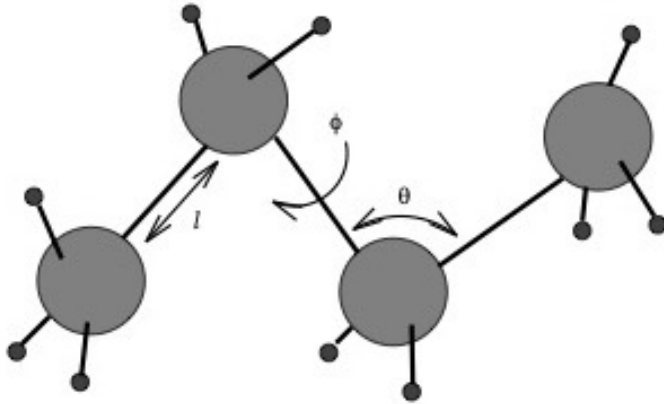
## LAMMPS commands

```
pair_style lj/cut 2.5  
pair_modify tail yes
```

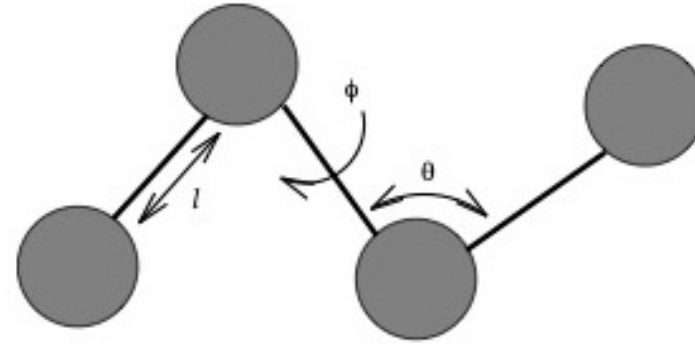
Can also correct pressure  
 $P_{LRC}$  is analytic

See Leach: pp 330-334

# Simulating multi-atom molecules



“all-atom” butane



“united-atom” butane

We need *intramolecular* force field terms to account for the molecule shape and motion

# Intramolecular terms:

## Bond lengths and angles

Bond length potentials often modeled as a harmonic

$$\mathcal{V}_b(\ell) = k_b(\ell - \ell_0)^2$$

force constant

nominal bond length

```
bond_style harmonic
```

```
bond_coeff 1 80.0 1.2 # type k_b l_0
```

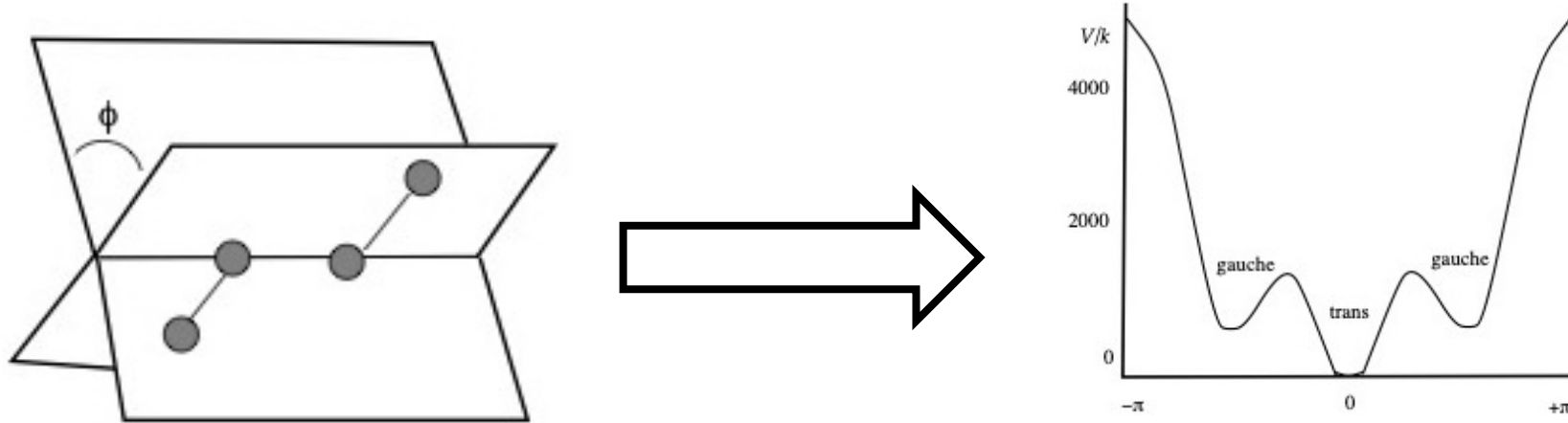
Harmonic bond angles

$$\mathcal{V}_\theta(\theta) = k_\theta(\theta - \theta_0)^2$$

```
angle_style harmonic
```

```
angle_coeff 1 300.0 107.0 # type k_theta theta0
```

# Dihedral angles



## Optimized Potentials for Liquid Simulation (OPLS) style

$$\mathcal{V}_\phi = \frac{1}{2}K_1[1 + \cos(\phi)] + \frac{1}{2}K_2[1 - \cos(2\phi)] + \frac{1}{2}K_3[1 + \cos(3\phi)] + \frac{1}{2}K_4[1 - \cos(4\phi)]$$

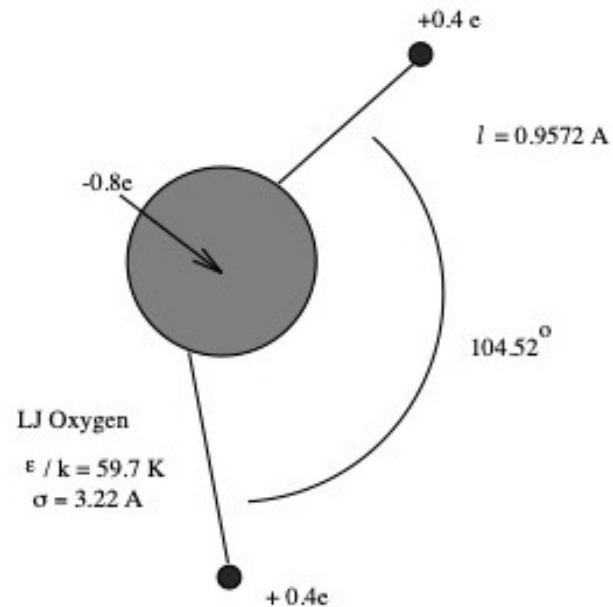
dihedral\_style opls

dihedral\_coeff 1 90.0 90.0 90.0 70.0

## Chemistry and HARvard Molecular Mechanics (CHARMM) style

$$\mathcal{V}_\phi = k_\phi[1 + \cos(n\phi - \delta)]$$

# Partial charges



Ex: Water (TIPS model)

- LJ + "partial charges" on atom centers
- Charges interact via Coulomb potential

$$\mathcal{V}_{zz}(r_{ij}) = \frac{z_i z_j}{4\pi\epsilon_0 r_{ij}}$$

Jorgensen, William L. (1981). "Quantum and statistical mechanical studies of liquids. 10. Transferable intermolecular potential functions for water, alcohols, and ethers. Application to liquid water". *Journal of the American Chemical Society*. American Chemical Society (ACS). **103** (2): 335–340. doi:[10.1021/ja00392a016](https://doi.org/10.1021/ja00392a016)

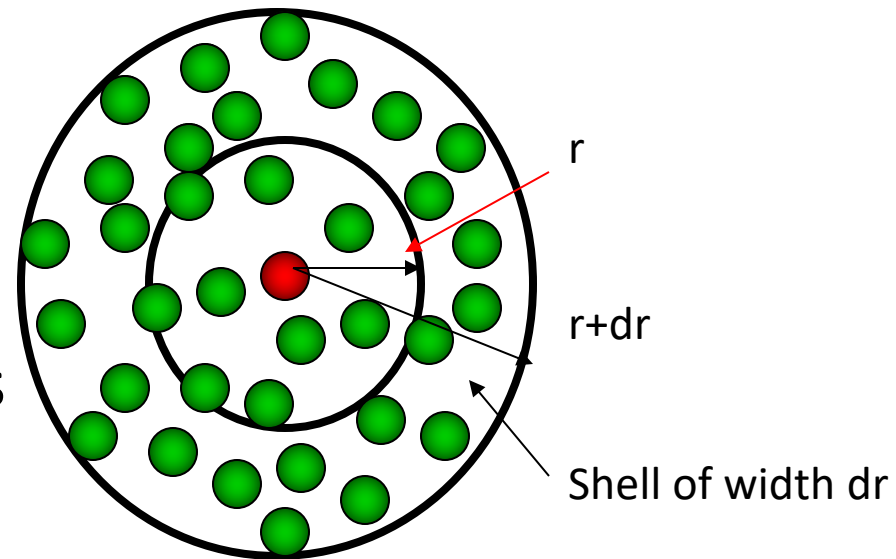
# Computing Properties

- What does a fluid “look” like?
- Imagine a snapshot of an atomic liquid

Fluid density in  
shell is

$$\rho(r)$$

- What are limits  
on  $\rho(r)$  ?



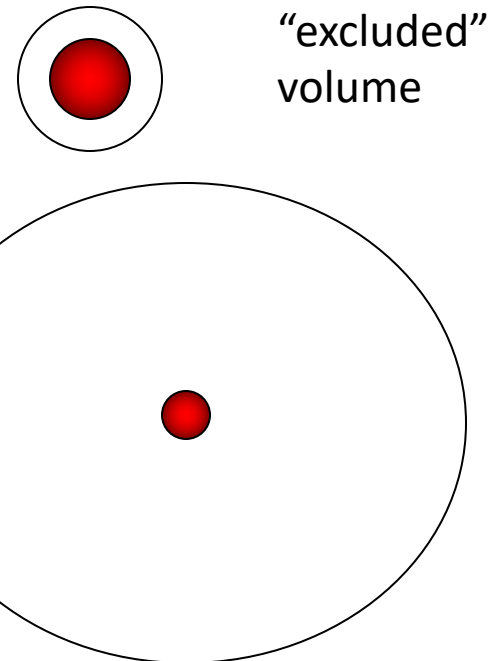
# Computing Properties

- What are limits on  $\rho(r)$  ?
- As  $r \rightarrow 0$ ?
- As  $r \rightarrow \text{infinity}$ ?

$$r \rightarrow 0, \rho(r) \rightarrow 0$$

$$r \rightarrow \infty, \rho(r) \rightarrow \rho$$

“bulk” fluid  
density





# Radial Distribution Function

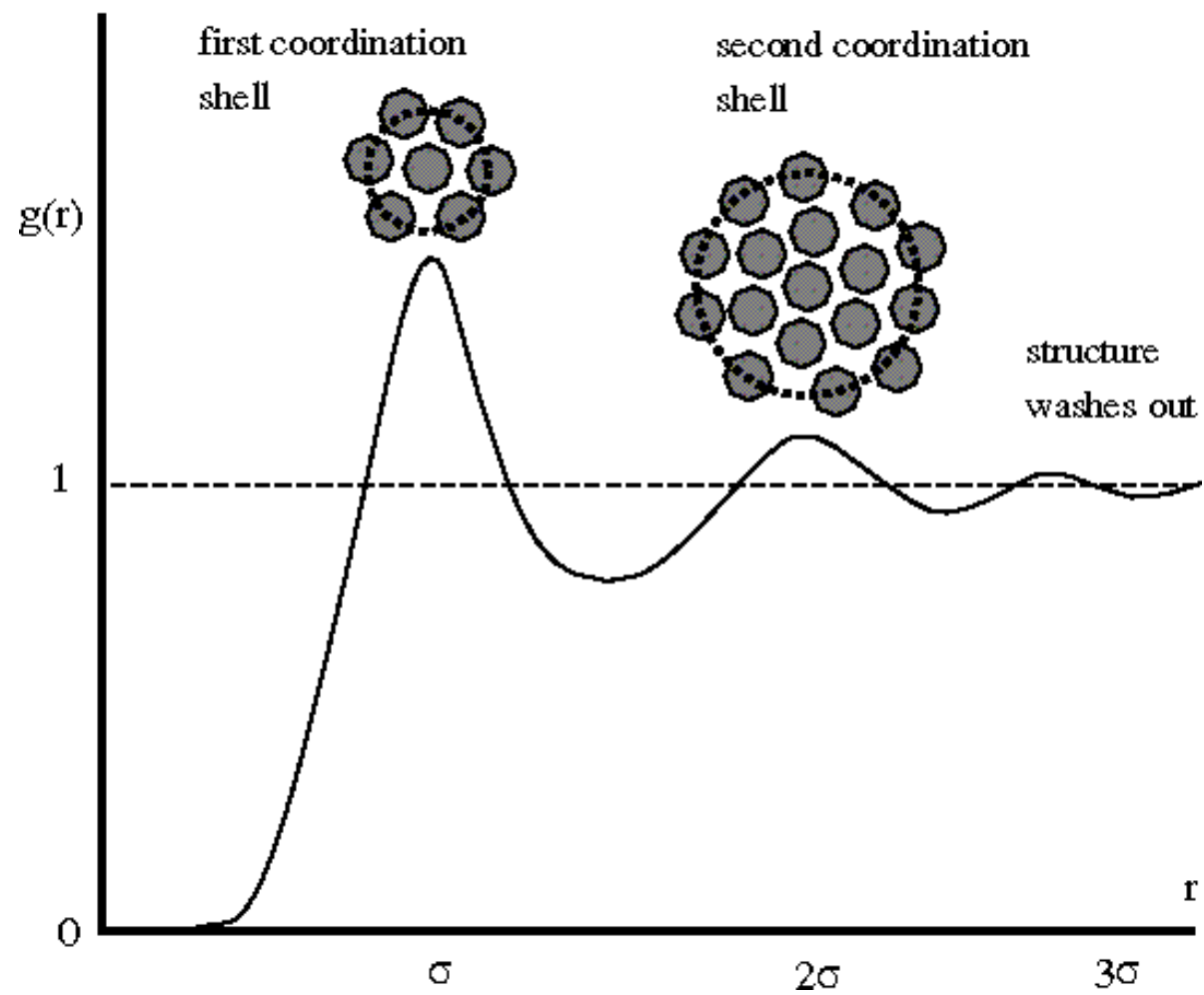
- First statement: fluids have short range “order”
- Last statement: fluids has no long range order.
- Define pair distribution function

$$g(r) \equiv \frac{\rho(r)}{\rho}$$

- Limits:

$$g(0) = 0, g(\infty) = 1$$

## *Liquid Structure*



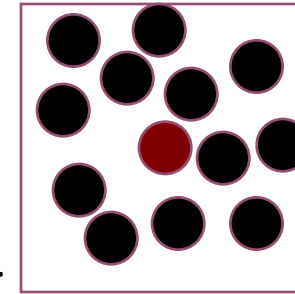
# Self-diffusivity

- Right-hand side is macroscopic property
  - applicable at macroscopic time scales

$$\langle r^2(t) \rangle = 2dDt$$

*Einstein  
equation*

- For any given configuration, each atom represents a point of high concentration (a weak fluctuation)



- View left-hand side of formula as the movement of this atom

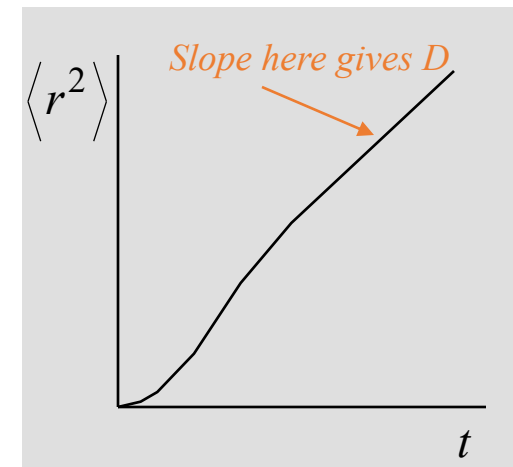
- ensemble average over all initial conditions

$$\langle r^2(t) \rangle = \int d\mathbf{p}^N \int d\mathbf{r}^N r_1^2(t) \delta(\mathbf{r}_1) \pi(\mathbf{r}^N, \mathbf{p}^N; t=0)$$

- asymptotic linear behavior of mean-square displacement gives diffusion constant

- independent data can be collected for each molecule

$$\langle r^2(t) \rangle = \frac{1}{N} \sum \langle r_i^2(t) \rangle$$



## PyLAT: Python LAMMPS Analysis Tools

Michael T. Humbert,<sup>✉</sup> Yong Zhang,<sup>✉</sup> and Edward J. Maginn<sup>\*✉</sup>

Department of Chemical and Biomolecular Engineering, University of Notre Dame, Notre Dame, Indiana 46556, United States

### PyLAT

This is the github repository for the Python LAMMPS Analysis Tools.

Some of the properties utilize Fortran for speedup. Before using this code, run either the shell script compile.sh or the python script compile.py

### Requirements

numpy>=1.14.1

scipy>=1.0.0

The following information can be accessed by running the command "python PyLAT.py -h"

usage:

```
PyLAT.py [-h] [-p PATH] [-i INPUT] [-f OUTPUT] [-g] [-c] [-m] [-d]
[--NEC] [--GKC] [--IPL] [--DEC] [--DS] [--Visc] --mol MOL
--nummol NUMMOL [-T TEMP] [-v VERBOSE] [--MSD_skip MSD_SKIP]
[--MSD_num_init MSD_NUM_INIT] [--D_Tolerance D_TOLERANCE]
[--RDF_timesteps RDF_TIMESTEPS] [--RDF_maxr RDF_MAXR]
[--RDF_binsize RDF_BINSIZE] [--GKC_skip GKC_SKIP]
[--GKC_Tolerance GKC_TOLERANCE] [--GKC_J_Output]
[--IPL_skip IPL_SKIP] [--DEC_start DEC_START]
[--DS_MOL1 DS_MOL1] [--DS_MOL2 DS_MOL2] [--DS_Dist DS_DIST]
[--DS_Dist_Tol DS_DIST_TOL] [--DS_First_Frame DS_FIRST_FRAME]
[--DS_Num_Samples DS_NUM_SAMPLES] [--Visc_Dir VISC_DIR]
[--Visc_Num VISC_NUM] [--Visc_Skip VISC_SKIP]
[--Visc_Num_Boots VISC_NUM_BOOTS]
[--Visc_Samples VISC_SAMPLES] [--Visc_Plot]
[--Visc_Guess VISC_GUESS VISC_GUESS VISC_GUESS VISC_GUESS]
LOG DAT TRJ [TRJ ...]
```

Analyze output from LAMMPS simulation

Required files are the log file, the data file and at least one trajectory file in that order

Also required is the name and number of each molecule type in the system. To do this, use the --mol and --nummol flags. Use the same order as they were entered into the data file.

# Building complex force fields for LAMMPS

- Generally, need to include a “data file” in addition to input file
  - Contains basic information about the size of the simulation, the initial atomic coordinates, molecular topology, and force field parameters.
- Lots of tools to build these files: we will go into detail about using MoSDeF but there are many other ways to do this

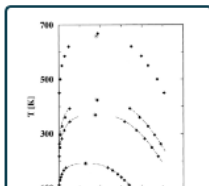
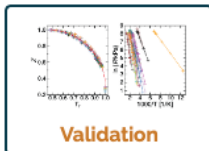
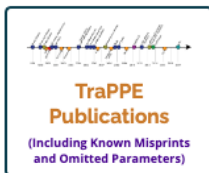
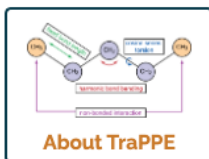
# Force field databases

<http://trappe.oit.umn.edu/>

## The Siepmann Group

Simulating Complex Chemical Systems

[TraPPE Home](#) [Validation](#)



## Tra<sub>n</sub>sferable P<sub>o</sub>tentials for P<sub>h</sub>ase E<sub>q</sub>uilibria

### About the TraPPE Force Field

The Transferable Potentials for Phase Equilibria family of force fields is a collection of functional forms and interaction parameters useful for modeling complex chemical systems with molecular mechanics simulation techniques. TraPPE maintains a high degree of accuracy in the prediction of thermophysical properties when applied to a range of different compounds, different state points, different compositions, and different properties. This makes TraPPE one of the few force fields generally suitable for materials and industrial applications.

The development of TraPPE models prioritizes **transferability** (maximizing the ability to build new chemical compounds by minimizing the number of (pseudo-) atoms needed) and **accuracy** (quantitative prediction of phase equilibria and other thermophysical properties over a wide range of physical conditions). Starting with "bonded" 1-2 (bond length), 1-3 (bond angle) and 1-4 (dihedral angle) interactions, TraPPE relies on experimental data or electronic structure calculations to provide equilibrium bond lengths and angles, the corresponding force constants, and the dihedral potentials for

### Search the TraPPE Database

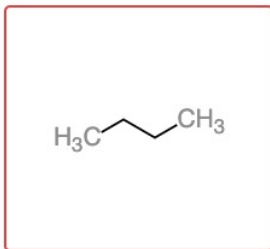
Enter a molecule name, formula or IUPAC InChI string:

Or, select from the complete list of available TraPPE models:

(choose molecule) ▼

Or, try building a molecule using the available TraPPE atom and interaction types.



## *n*-butane (UA)

**Chemical** C<sub>4</sub>H<sub>10</sub>

**Formula:**

**Molecular** **58.12**

**Weight:**

**Smiles String:** CCCC

InChI=1S/C4H10/c1-3-4-2/h3-4H2,1-2H3

This model was assembled from previously parameterized TraPPE building blocks and was tested for accuracy as part of the transferability verification process.

Parameters

Simulation Data

Downloads

### Functional Forms and Parameters

#### Nonbonded Interactions

$$u_{\text{NB}}(r_{ij}) = 4\epsilon_{ij} \left[ \left( \frac{\sigma_{ij}}{r_{ij}} \right)^{12} - \left( \frac{\sigma_{ij}}{r_{ij}} \right)^6 \right] + \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}}$$

$$\sigma_{ij} = \frac{1}{2}(\sigma_{ii} + \sigma_{jj}) \quad \epsilon_{ij} = (\epsilon_{ii}\epsilon_{jj})^{1/2}$$

#	(pseudo)atom	type	$\epsilon/k_B$ [K]	$\sigma$ [Å]	$q$ [e]
1	CH3	[CH3]-CHx	98	3.75	0.0
2	CH2	CHx-[CH2]-CHx	46	3.95	0.0
3	CH2	CHx-[CH2]-CHx	46	3.95	0.0
4	CH3	[CH3]-CHx	98	3.75	0.0

#### 1-2 Bonded Interactions

(TraPPE uses fixed bond lengths)

#	Stretch	Type	Length [Å]
1	1 - 2	CHx-CHy	1.54
2	2 - 3	CHx-CHy	1.54
3	3 - 4	CHx-CHy	1.54

#### 1-3 Bonded Interactions

$$u_{\text{bend}}(\theta) = \frac{k_\theta}{2} (\theta - \theta_{\text{eq}})^2$$

#	Bend	Type	$\theta$ [°]	$k_\theta/k_B$ [K/rad <sup>2</sup> ]
1	1 - 2 - 3	CHx-(CH2)-CHy	114	62500
2	2 - 3 - 4	CHx-(CH2)-CHy	114	62500

#### 1-4 Bonded Interactions

(Read more about TraPPE dihedral conventions)

$$u_{\text{torsion}}(\phi) = c_0 + c_1 [1 + \cos(\phi)] + c_2 [1 - \cos(2\phi)] + c_3 [1 + \cos(3\phi)]$$

#	Torsion	Type	$c_0/k_B$ [K]	$c_1/k_B$ [K]	$c_2/k_B$ [K]	$c_3/k_B$ [K]
1	1 - 2 - 3 - 4	CHx-(CH2)-(CH2)-CHy	0.00	355.03	-68.19	791.32

```
# Example of beginning input file using data file
```

```
variable      NAME index methane
```

```
log           ${NAME}.log
```

```
variable      infile index methane.data
```

```
variable      mytemp index 328.15
```

```
# set up simulation
```

```
units         real
```

```
atom_style    full
```

```
boundary      p p p
```

```
pair_style    lj/cut/coul/long 12
```

```
kspace_style  ewald 0.0001
```

```
pair_modify   tail yes
```

```
pair_modify   mix arithmetic
```

```
special_bonds amber
```

```
bond_style    harmonic
```

```
angle_style   harmonic
```

```
#dihedral_style opls
```

```
#improper_style cvff
```

```
read_data     ${infile}
```



## LAMMPS Description (1st line of file)

100 atoms (this must be the 3rd line, 1st 2 lines are ignored)  
95 bonds (# of bonds to be simulated)  
50 angles (include these lines even if number = 0)  
30 dihedrals  
20 impropers  
5 atom types (# of nonbond atom types)  
10 bond types (# of bond types = sets of bond coefficients)  
18 angle types  
20 dihedral types (do not include a bond,angle,dihedral,improper type  
2 improper types line if number of bonds,angles,etc is 0)  
-0.5 0.5 xlo xhi (for periodic systems this is box size, -  
0.5 0.5 ylo yhi for non-periodic it is min/max extent of atoms)  
-0.5 0.5 zlo zhi (do not include this line for 2-d simulations)

### Masses

1 mass ... N mass (N = # of  
atom types)

### Nonbond Coeffs

1 coeff1 coeff2 ... ... N coeff1 coeff2 ... (N = # of atom types)

methane.data - created by mBuild; units = real

2500 atoms

2000 bonds

3000 angles

0 dihedrals

0 impropers

2 atom types

1 bond types

1 angle types

0.000000 40.000000 xlo xhi

0.000000 40.000000 ylo yhi

0.000000 40.000000 zlo zhi

Masses

1	12.010780	# opls_138
---	-----------	------------

2	1.007947	# opls_140
---	----------	------------

Pair Coeffs # lj

#	epsilon (kcal/mol)		sigma (Angstrom)	
1	0.06600	3.50000		# opls_138
2	0.03000	2.50000		# opls_140

Bond Coeffs # harmonic

#	k(kcal/mol/angstrom^2)		req(angstrom)		
1	340.0	1.09		# opls_138	opls_140

Angle Coeffs # harmonic

#	k(kcal/mol/rad^2)		theteq(deg)		
1	33.0	107.80000		# opls_140	opls_138    opls_140

Atoms # full

1	1	1	-0.240000	36.169297	32.738712	18.053636
2	1	2	0.060000	35.618914	31.986198	17.452327
3	1	2	0.060000	35.460444	33.509299	18.420370
4	1	2	0.060000	36.945195	33.219973	17.423444
5	1	2	0.060000	36.652636	32.239379	18.918402
6	1	1	-0.240000	23.235838	22.713311	15.456040
7	1	2	0.060000	22.294018	22.608173	14.879246
8	1	2	0.060000	23.222708	23.672111	16.013982
9	1	2	0.060000	24.099230	22.702595	14.759470
10	1	2	0.060000	23.327397	21.870364	16.171462
11	1	1	-0.240000	9.485563	13.833860	7.944599
12	1	2	0.060000	8.658834	13.469003	8.588158
13	1	2	0.060000	9.944858	12.976459	7.411044
14	1	2	0.060000	10.251984	14.331392	8.573749
15	1	2	0.060000	9.086575	14.558589	7.205445
16	1	1	-0.240000	31.850319	18.437411	36.362290
17	1	2	0.060000	31.507804	19.006143	37.251108
18	1	2	0.060000	32.687095	18.979813	35.876105
19	1	2	0.060000	31.011002	18.332433	35.644448
20	1	2	0.060000	32.195375	17.431255	36.677498

# OPLS – “pre-built”

leelasd / OPLSAA-DB Public

Notifications Fork 4 Star 19

<> Code Issues Pull requests Actions Projects Security Insights

master 1 branch 0 tags Go to file Code

leelasd Added All the Files Needed		b09d51b on Aug 2, 2018 2 commits
NEGATIVE_CHARGED_MOLECULES	Added All the Files Needed	4 years ago
NEUTRAL_MOLECULES	Added All the Files Needed	4 years ago
POSITIVE_CHARGED_MOLECULES	Added All the Files Needed	4 years ago
ZMATS	Added All the Files Needed	4 years ago
CHARGE_DAT	<a href="#">Added All the Files Needed</a>	4 years ago
LICENSE	<a href="#">Added All the Files Needed</a>	4 years ago
MOLECULES_woPLSAA.csv	Added All the Files Needed	4 years ago
README.md	Initial commit	4 years ago

README.md

## OPLSAA-DB

Database of OPLS-AA parameters and topologies for 464 molecules. Zip files contains parameter and topologies for OpenMM, Gromacs, NAMD, CHARMM, LAMMPS, TINKER, CNS/X-PLOR, Q, DESMOND, BOSS and MCPRO.

### About

Database of hand-built OPLS-AA parameters and topologies for 464 molecules. Zip files contains parameter and topologies for OpenMM, Gromacs, NAMD, CHARMM, LAMMPS, TINKER, CNS/X-PLOR, Q, DESMOND, BOSS and MCPRO.

[traken.chem.yale.edu/ligpargen/](https://traken.chem.yale.edu/ligpargen/)

molecular-dynamics openmm molecular-simulation thermodynamics lammps charmm force-field namd gromacs tinker opls-aa

Readme Apache-2.0 license 19 stars 4 watching 4 forks

### Releases

No releases published

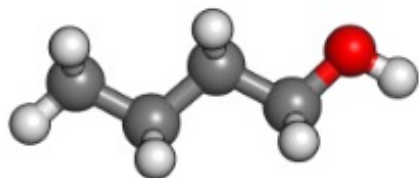
# Build your own OPLS

An easy to use tool

<http://zarbi.chem.yale.edu/ligpargen/>

1. Enter SMILES code: example

OCCCC = 1-butanol



2. Create PDB file (can do this with many other tools such as Avogadro)

```
REMARK LIGPARGEN GENERATED PDB FILE
ATOM 1 O00 UNK 1 1.000 1.000 0.000
ATOM 2 C01 UNK 1 -0.421 1.000 0.000
ATOM 3 C02 UNK 1 -0.911 1.000 1.441
ATOM 4 C03 UNK 1 -2.436 0.998 1.516
ATOM 5 C04 UNK 1 -2.922 1.000 2.956
ATOM 6 H05 UNK 1 1.284 0.999 -0.930
ATOM 7 H06 UNK 1 -0.768 0.108 -0.531
ATOM 8 H07 UNK 1 -0.768 1.892 -0.531
ATOM 9 H08 UNK 1 -0.506 1.877 1.962
ATOM 10 H09 UNK 1 -0.504 0.125 1.962
ATOM 11 H0A UNK 1 -2.832 0.114 1.004
ATOM 12 H0B UNK 1 -2.834 1.880 1.001
ATOM 13 H0C UNK 1 -2.566 0.114 3.490
ATOM 14 H0D UNK 1 -4.017 0.997 2.985
ATOM 15 H0E UNK 1 -2.568 1.889 3.487
TER
CONNECT 1 2
CONNECT 2 3
CONNECT 3 4
CONNECT 4 5
CONNECT 1 6
CONNECT 2 7
CONNECT 2 8
CONNECT 3 9
CONNECT 3 10
CONNECT 4 11
CONNECT 4 12
CONNECT 5 13
CONNECT 5 14
CONNECT 5 15
END
```

# Build your own OPLS

- LigParGen will spit out a LAMMPS datafile for a single molecule in addition to the pdb (and other “data files” for different packages)
- Partial charges computed using one of two methods
- You should always verify the parameters you get!
- A huge source of error and inconsistency in the literature is incorrect force field parameters.
- Lots of other resources

<https://github.com/paduagroup/fftool>

(inconsistent with LigParGen but conversion tools available)

# What about other force fields?

- Lots of tools and converters to go between formats
- Eventually, you will likely need to write or modify scripts
- Great care is required because it is easy to make mistakes!
- For all but the simplest molecules, write scripts
- We will show you how to use a general tool under development called MoSDeF

# Molecular Simulation Design Framework (MoSDeF)

MoSDeF presents a set of extensible Python tools designed to facilitate the initialization, atom-typing, and screening of soft matter systems using molecular simulation.

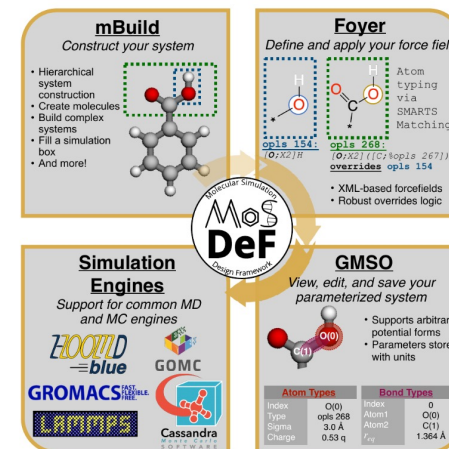
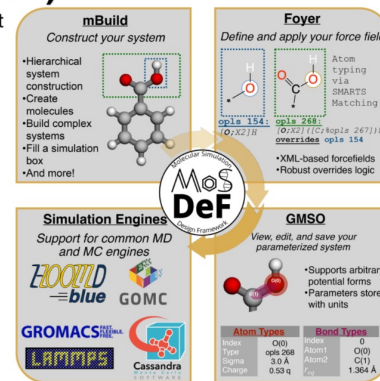
GET STARTED!

The MoSDeF tools have been designed to facilitate the hierarchical construction of both atomistic and coarse-grained system configurations, the application of classical force fields (i.e., atom-typing), encapsulation of chemical topology information, and the generation of input files for a variety of molecular dynamics and Monte Carlo simulation engines, including LAMMPS, GROMACS, HOOMD-Blue, Cassandra, and GOMC.



## The Molecular Simulation Design Framework (MoSDeF)

- ❑ MoSDeF is a collection of Python libraries to assist in the system initialization stage of molecular simulation in a reproducible manner
  - Developed with best practices of OSS dev
    - Test driven development, version control,
- ❑ **mBuild**<sup>1</sup> - A generic molecule builder
  - Build systems up from smaller re-usable components
  - Plug-in system for users to share and extend complex systems
- ❑ **Foyer**<sup>2</sup> - Apply interaction parameters in a simulation engine-agnostic manner
  - Extends OpenMM forcefield XMLs
  - Plug-in system for users to share forcefields
- ❑ **GMSO** - Molecular topology information
  - Replacing ParmEd
  - Interaction parameters and functional forms



<sup>1</sup> Klein C, Sallal J, Jones TJ, Iacovella CR, McCabe C, Cummings PT. Foundations of Molecular Modeling and Simulation. 2016.  
<sup>2</sup> Klein C, Summers AZ, Thompson MM, Gilmer JB, McCabe C, Cummings PT, Sallal J, Iacovella CR. Computational Materials Science. 161; 2019

<https://github.com/mosdef-hub>

[mosdef.org](https://mosdef.org)