CyberMAGICS Workshop RXMD Hands-on Session

Ken-ichi Nomura, Marco Olguin, Tian Sang, Nitish Baradwaj, Pranab Sarker

June 6, 2025

Outline

- Create Initial Configuration
- RXMD Input Parameters
- Hands-on : MoO_3 Self Reduction

RXMD Hands-on: Software Setup

- Unzip RXMD code

```
$ unzip rxmd-cybermagics.zip
```

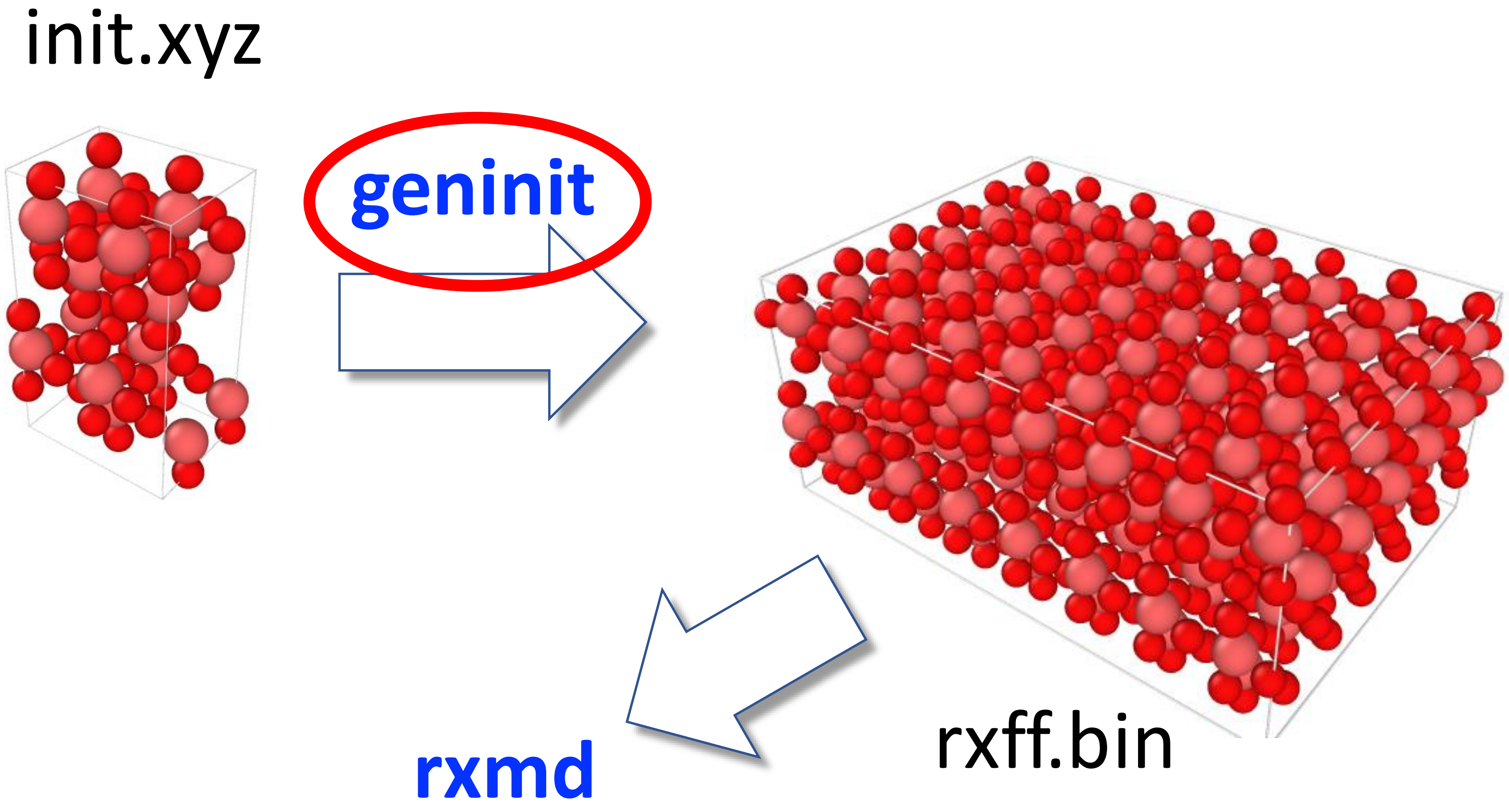
RXMD directory structure looks like this.

```
.
├── DAT
│   └── rxff.bin
├── ffield
├── init
│   ├── geninit.F90
│   ├── input.xyz
│   └── Makefile
├── Makefile.inc
├── rxmd.in
└── src
```

```
src/
├── bo.F90
├── cg.F90
├── comm.F90
├── fileio.F90
├── init.F90
├── main.F90
├── Makefile
├── module.F90
├── param.F90
├── pot.F90
├── qeq.F90
├── stress
└── stress.F90
```

Outline

- Create Initial Configuration
- RXMD Input Parameters
- Hands-on : MoO_3 Self Reduction


init.xyz
geninit
rxff.bin
rxmd

Create Initial Configuration : geninit

- We use an executable called **geninit** (**gen**erate **init**ial config) to generate initial configuration for RXMD simulation.
- **geninit** reads unit cell information from **input.xyz** (by default) and ReaxFF force field file (**../ffield**) to find numerical IDs from element name (for example C (carbon) is 1, H (hydrogen) is 2), then creates a binary file **rxff.bin**, input file for RXMD.
- To build **geninit**, go to **init** directory and type **make**.

```
$ cd init
$ make
```

```
--------------------------------------------------------------
        input file: input.xyz
       ffield file: ../ffield
    nprocs,vprocs:         1     1     1     1
         mctot,mc:         6     2     3     1
--------------------------------------------------------------
…
```

Create Initial Configuration : geninit

- geninit command takes several options

```
$ ./geninit -help
./geninit -mc 1 1 1 -vprocs 1 1 1 -inputxyz
input.xyz -ffield ffield [-r or -n]
```

-mc or -m (3 integers) : Number of repetitions of unit cell.

-vprocs or -v (3 integers) : Number of processors in x,y, and z directions

-inputxyz or -i (string) : Filename contains unit cell configuration

-ffield or -f (string) : Filename contains ReaxFF force field parameters

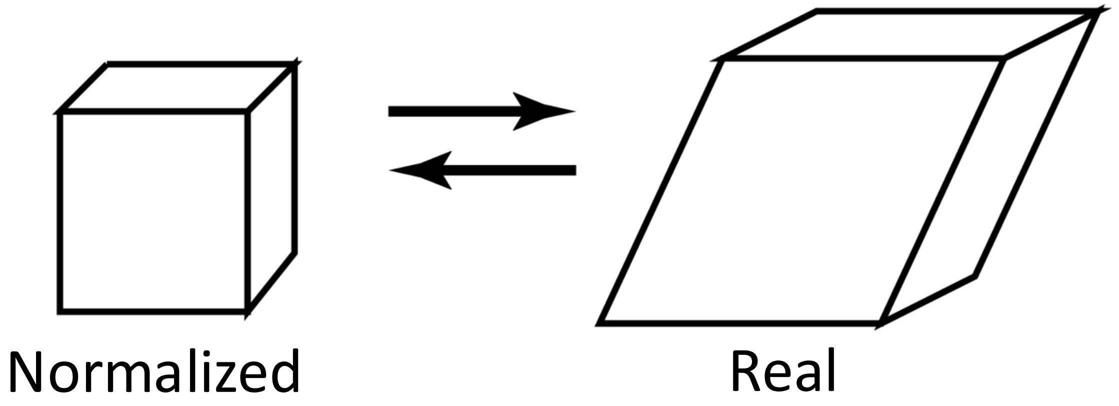
Create Initial Configuration : geninit

- **geninit** supports normalized and real coordinate conversion.

```
$ ./geninit -help
./geninit -mc 1 1 1 -vprocs 1 1 1 -inputxyz
input.xyz -ffield ffield [-r or -n]
```

-getreal or -r : Convert from normalized to real coordinates. Result will be stored in **real.xyz**.

-getnorm or -n : Convert from real to normalized coordinates. Result will be stored in **norm.xyz**.



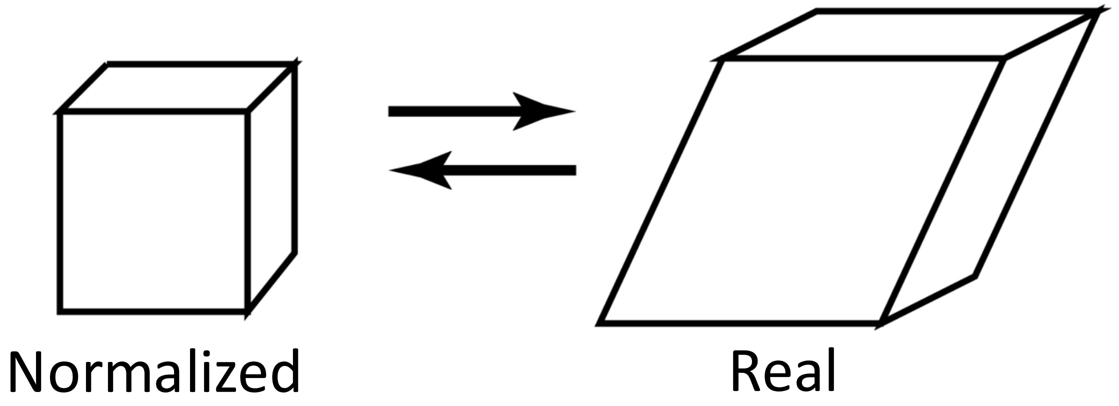
Create Initial Configuration : geninit

- **geninit** supports normalized and real coordinate conversion.

```
$ ./geninit -help
./geninit -mc 1 1 1 -vprocs 1 1 1 -inputxyz
input.xyz -ffield ffield [-r or -n]
```

- **-r** and **-n** flags can be used together with **–i** to specify input file name and –**mc** to repeat the unit structure but **-v** will be ignored.

Caveat! There is no check on the coordinates of input data. It is the user's responsibility to provide proper input coordinate data.



Create Initial Configuration : input.xyz

- Input file **input.xyz** resembles XYZ format but is slightly modified.
- **Line1 :** number of atoms in unit cell followed by a string to describe the unit cell.
- **Line2 :** six lattice parameters, *a, b, c* and *alpha*, *beta*, and *gamma*.

```
64 "MoO3 unit cell"
7.92   7.39  13.86  90.00  90.00  90.00
Mo 0.141162 0.137258 0.354299
…
O 0.0982146 0.62335 0.187911
```

Create Initial Configuration : input.xyz

- Line3-EOF : element name and x, y, and z positions.

Caveat! element name must exist in ReaxFF force field file.

Caveat! atom coordinate are normalized by the lattice parameters.

```
64 "MoO3 unit cell"
7.92   7.39  13.86  90.00  90.00  90.00
Mo 0.141162 0.137258 0.354299
…
O 0.0982146 0.62335 0.187911
```

Outline

- Create Initial Configuration
- RXMD Input Parameters
- Hands-on : MoO_3 Self Reduction

RXMD Input Parameters : rxmd.in

- When RXMD executable is invoked, it reads **rxmd.in** for various simulation-related parameters.

```
$ cat rxmd.in
```

```
mdmode          1                  #<mdmode>
time            0.25     5000      #<dt> <ntime_step>
temperature     300  1.0   100     #<treq> <vsfact> <sstep>
io_step         1000     100       #<fstep>  <pstep>
processors      1 1 1              #<vprocs>
Qeq             1  500  1.d-6  1   #<isQEq> <NMAXQEq> <QEq_tol> <qstep>
Io_type .true. .true.  .true.      #<isBinary> <isBondFile> <isPDB>
CG_tol  1.d-8                      #<ftol>
```

Keyword   Variable values   Variable names in program

RXMD Input Parameters : rxmd.in

- **mdmode** decides overall behavior of RXMD simulation.
- **mdmode** = 1 is NVE run, 4-7 are various temperature control modes by velocity scaling, and 10 for structural optimization using conjugate gradient method.

```
mdmode         1                #<mdmode>
time           0.25     5000    #<dt> <ntime_step>
temperature    300  1.0   100   #<treq> <vsfact> <sstep>
io_step        1000     100     #<fstep>  <pstep>
processors     1 1 1            #<vprocs>
Qeq            1  500  1.d-6  1 #<isQEq> <NMAXQEq> <QEq_tol> <qstep>
Io_type .true. .true. .true.    #<isBinary> <isBondFile> <isPDB>
CG_tol  1.d-8                   #<ftol>
```

RXMD Input Parameters : rxmd.in

- **dt** is one MD timestep in femtosecond unit. e.g. 0.25 = 0.25(fs)
- **ntime_step** is the number of MD steps to run.

```
mdmode          1                 #<mdmode>
time            0.25     5000     #<dt> <ntime_step>
temperature     300  1.0   100    #<treq> <vsfact> <sstep>
io_step         1000     100      #<fstep>  <pstep>
processors      1 1 1             #<vprocs>
Qeq             1  500  1.d-6  1  #<isQEq> <NMAXQEq> <QEq_tol> <qstep>
Io_type .true. .true. .true.      #<isBinary> <isBondFile> <isPDB>
CG_tol  1.d-8                     #<ftol>
```

RXMD Input Parameters : rxmd.in

- When mdmode == 4, atom velocity is multiplied by **vsfact** every **sstep** MD steps.
- **treq** is not used with mdmode == 4.
- **sstep** is the interval of each velocity scaling, e.g. **sstep** == 100 means velocity scaling every100 MD steps.

```
mdmode          4                  #<mdmode>
time            0.25     5000      #<dt> <ntime_step>
temperature     300  1.0   100     #<treq> <vsfact> <sstep>
io_step         1000     100       #<fstep>  <pstep>
processors      1 1 1              #<vprocs>
Qeq             1  500  1.d-6  1   #<isQEq> <NMAXQEq> <QEq_tol> <qstep>
Io_type .true. .true. .true.       #<isBinary> <isBondFile> <isPDB>
CG_tol  1.d-8                      #<ftol>
```

RXMD Input Parameters : rxmd.in

- **treq** is used when mdmode == 5, 6 and 7 where atom velocity is scaled to **treq** (K) every **sstep** MD steps.
- **sstep** is the interval of each velocity scaling, e.g. **sstep** == 100 means velocity scaling every100 MD steps.

```
mdmode          5                 #<mdmode>
time            0.25     5000     #<dt> <ntime_step>
temperature     300   1.0   100   #<treq> <vsfact> <sstep>
io_step         1000     100      #<fstep>  <pstep>
processors      1 1 1             #<vprocs>
Qeq             1  500  1.d-6  1  #<isQEq> <NMAXQEq> <QEq_tol> <qstep>
Io_type .true. .true. .true.      #<isBinary> <isBondFile> <isPDB>
CG_tol  1.d-8                     #<ftol>
```

RXMD Input Parameters : rxmd.in

- **fstep** is the interval of check-pointing, i.e. save atom data and connectivity data on to disk. Type of data to be saved is determined by **isBinary**, **isBondFile**, **isPDB,** and **isXYZ** logical variables.
- **pstep** is the interval of displaying ReaxFF energy terms on standard output.

```
mdmode          1                  #<mdmode>
time            0.25    5000       #<dt> <ntime_step>
temperature     300  1.0   100     #<treq> <vsfact> <sstep>
io_step         1000    100        #<fstep>  <pstep>
processors      1 1 1              #<vprocs>
Qeq             1  500  1.d-6  1   #<isQEq> <NMAXQEq> <QEq_tol> <qstep>
Io_type         .true. .true. .false. .true. #<isBinary> <isBondFile>
                                              <isPDB> <isXYZ>
CG_tol  1.d-8                      #<ftol>
```

RXMD Input Parameters : rxmd.in

- **vprocs** is the number of processors in x, y, and z directions, dividing the total simulation box into smaller subdomains.

  Caveat! **vprocs** must be either 1 or even number.

```
mdmode          1                 #<mdmode>
time            0.25     5000     #<dt> <ntime_step>
temperature     300  1.0   100    #<treq> <vsfact> <sstep>
io_step         1000    100       #<fstep>  <pstep>
processors      1 1 1             #<vprocs>
Qeq             1  500  1.d-6  1  #<isQEq> <NMAXQEq> <QEq_tol> <qstep>
Io_type         .true. .true. .false. .true. #<isBinary> <isBondFile>
                                              <isPDB> <isXYZ>
CG_tol  1.d-8                     #<ftol>
```

RXMD Input Parameters : rxmd.in

- **isQEq** is a logical flag to enable the variable charge (isQEq == 1) or disable it (isQEq == 0).
- QEq minimize the electrostatic energy using conjugate gradient algorithm. **NMAXQEq**, **Qeq_tol**, and **qsteps** are the maximum number of iteration, the convergence tolerance and interval of QEq subroutine call, respectively.

```
mdmode          1                 #<mdmode>
time            0.25    5000      #<dt> <ntime_step>
temperature     300  1.0   100    #<treq> <vsfact> <sstep>
io_step         1000    100       #<fstep>  <pstep>
processors      1 1 1             #<vprocs>
Qeq             1  500  1.d-6  1  #<isQEq> <NMAXQEq> <QEq_tol> <qstep>
Io_type         .true. .true. .false. .true. #<isBinary> <isBondFile>
                                              <isPDB> <isXYZ>
CG_tol  1.d-8                     #<ftol>
```

RXMD Input Parameters : rxmd.in

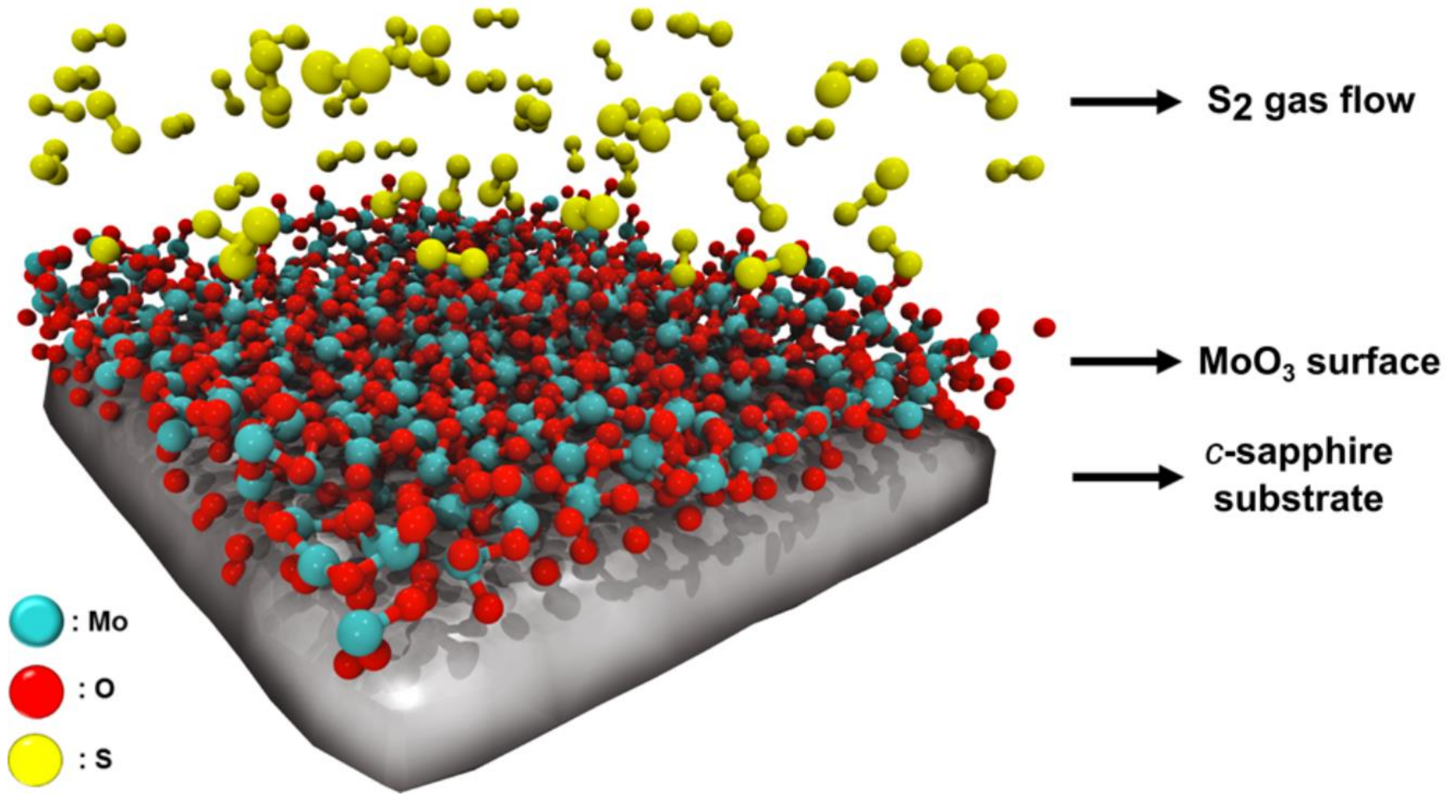
- **ftol** is the tolerance of conjugate gradient for structural optimization. Not for charge QEq.
- **ftol** is used when **mdmode** == 10.

```
mdmode          1                  #<mdmode>
time            0.25     5000      #<dt> <ntime_step>
temperature     300  1.0   100     #<treq> <vsfact> <sstep>
io_step         1000    100        #<fstep>  <pstep>
processors      1 1 1              #<vprocs>
Qeq             1  500  1.d-6  1   #<isQEq> <NMAXQEq> <QEq_tol> <qstep>
Io_type         .true. .true. .false. .true. #<isBinary> <isBondFile>
                                              <isPDB> <isXYZ>
CG_tol  1.d-8                      #<ftol>
```

Outline

- Create Initial Configuration
- RXMD Input Parameters
- Hands-on : MoO_3 Self Reduction

RXMD Hands-on : MoO_3 Self-Reduction Simulation



Computational synthesis of MoS2 layers by reactive molecular dynamics simulations, initial sulfidation of MoO3 surfaces S. Hong, et al. *Nano Letters* **17**, 4866-4872 (2017)

RXMD Hands-on : MoO_3 Self-Reduction Simulation

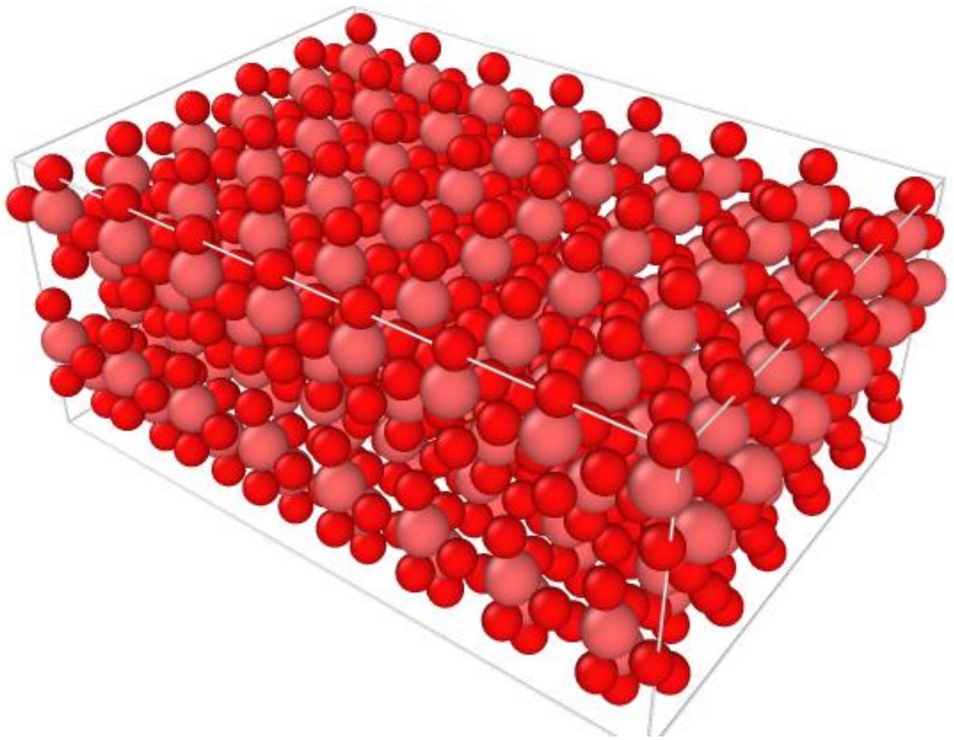
- Change directory to **init.moo3** and type **make** to create initial config.

```
$ cd   init.moo3/
$ gfortran geninit.F90 –o gfortran
$ ./geninit -i input.xyz
$ cp -v rxff.bin ../DAT
```

```
gfortran -c   geninit.F90
gfortran -o geninit geninit.o
./geninit input.xyz
 input file:   input.xyz
 ffield file:   ../ffield
 nprocs,vprocs          1     1     1     1
 mctot,mc         12      4     3     1
  1-O   2-S   3-Mo  4-Al
        64   MoO3 unit cell
...
...
cp -v rxff.bin ../DAT
'rxff.bin' -> '../DAT/rxff.bin'
```

RXMD Hands-on : MoO_3 Self-Reduction Simulation

- The system looks like this,



- Number of Atoms : 768
192 Mo + 576 O
- Lattice Parameters:
31.68(Å)x22.17(Å)x41.58(Å)
90.0 90.0 90.0
- 30 (Å) vacuum in z-axis
- Relax free surface and heatup the system upto 1800(K)

Simulation Schedule

- First we relax the free surfaces by quenching, then increase the system temperature up to 1800K by velocity scaling.
- Simulation schedule and input parameters are following.

1. Surface Relaxation :

rxmd.in-00 : for 1000 MD steps
rxmd.in-01 : for 1000 MD steps
rxmd.in-02 : for 1000 MD steps

2. Heatup :

rxmd.in-03 : to 600K for 5000 MD steps
rxmd.in-04 : to 1200K for 5000 MD steps
rxmd.in-05 : to 1800K for 5000 MD steps

13. Measurement :

Keep temperature at 1800K and run.

Step 1: 01-relax.sh

rxmd.in-00

```
mdmod             4
time              0.01    1000
Temperature       100  0.5   100
io_step           100     100
Processors        1 1 1
QEq               1  500  1.d-6  10
io_type           .true. .true. .false. .true.
CG_tol 1.d-8
```

rxmd.in-01

```
4
0.5 1000
100 0.5 100
100 100
1 1 1
1 500 1.d-6 10
1.0 180
.true. .true. .true.
1.d-8
```

rxmd.in-02

```
4
0.5 1000
100 0.9 100
100 100
1 1 1
1 500 1.d-6 10
1.0 180
.true. .true. .true.
1.d-8
```

Step 2 & 3 : 02-heatup.sh & 03-run.sh

rxmd.in-03

```
7                        <mdmod>
0.5    5000            <dt> <ntime_step>
600  0.9   100         <treq> <vsfact> <sstep>
100     100             <fstep>  <pstep>
1 1 1                   <vprocs>
1  500  1.d-6  10       <isQEq> <NMAXQEq> <QEq_tol> <qstep>
1.0  180                <Lex_fqs> <Lex_k>
.true. .true. .true. <isBinary> <isBondFile> <isPDB>
1.d-8                   <ftol>
```

rxmd.in-04

```
7
0.5 5000
1200 0.9 100
100 100
1 1 1
1 500 1.d-6 10
1.0 180
.true. .true. .true.
1.d-8
```

rxmd.in-05

```
5
0.5 5000
1800 0.9 100
100 100
1 1 1
1 500 1.d-6 10
1.0 180
.true. .true. .true.
1.d-8
```

Analyze Simulation Result : Visualize Atom Trajectory

- While your job is running, checkpoint data (.bin), atom trajectory (.pdb), and connectivity information (.bnd) will be saved into **DAT** directory.

```
$ ls DAT/
000000000.bin
000000000.bnd
000000000.pdb
000000100.bin
000000100.bnd
000000100.pdb
...
```

- To visualize atom trajectory with VMD, we need to concatenate PDB files from different MD steps into one PDB file with a proper separator keyword [**END**].
- Also, every line must have the same atom through all MD frames.

Analyze Simulation Result : Bond Analysis

- A simple Python script **count_bond.py** is included in the tarball.
- **count_bond.py** counts the number of bonds of each bond type.
- No argument is necessary, just run **count_bond.py** from your working directory that has **DAT** directory.

```
$ ./count_bond.py
```

- You will see output below.

```
...
./DAT/000080000.bnd  :  1-1   22   1-3   2092   3-3   42
./DAT/000080100.bnd  :  1-1   22   1-3   2124   3-3   36
./DAT/000080200.bnd  :  1-1   22   1-3   2132   3-3   42
./DAT/000080300.bnd  :  1-1   22   1-3   2120   3-3   34
./DAT/000080400.bnd  :  1-1   22   1-3   2154   3-3   36
...
```

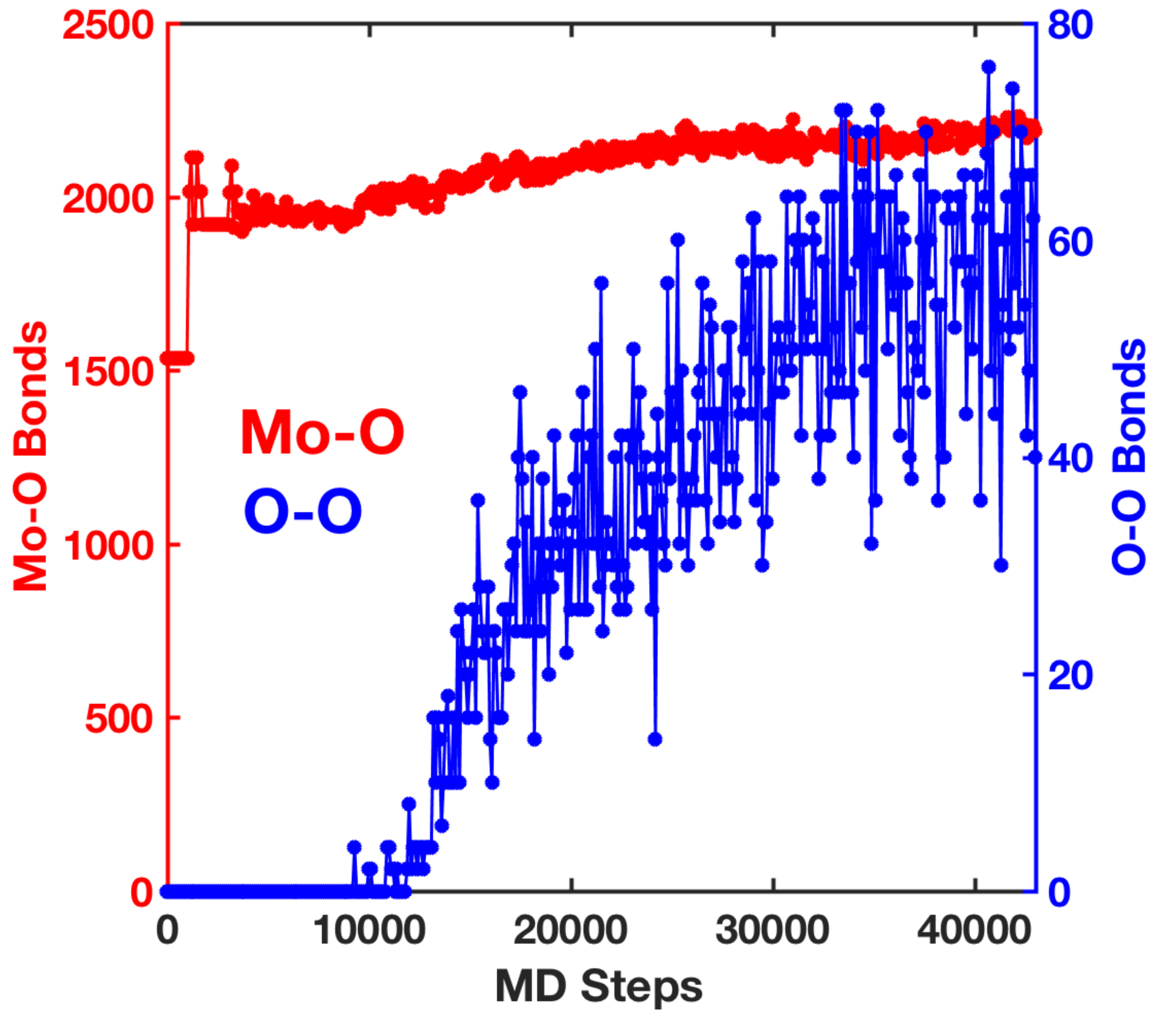
Analyze Simulation Result : Bond Analysis

- Blue columns are atom type combinations, e.g. 1-Mo and 3-O, and red columns are the number of bonds.

```
...
./DAT/000080000.bnd  :  1-1   22   1-3   2092   3-3   42
./DAT/000080100.bnd  :  1-1   22   1-3   2124   3-3   36
./DAT/000080200.bnd  :  1-1   22   1-3   2132   3-3   42
./DAT/000080300.bnd  :  1-1   22   1-3   2120   3-3   34
./DAT/000080400.bnd  :  1-1   22   1-3   2154   3-3   36
./DAT/000080500.bnd  :  1-1   22   1-3   2118   3-3   46
...
```

- Use any software to plot the number of bonds for each bond-type.

Analyze Simulation Result : Bond Analysis


Mo-O
O-O
Mo-O Bonds
O-O Bonds
MD Steps