

2017年9月7日(木) 計算科学振興財団

LAMMPSでの仮想実験セミナー 架橋ゴムの応力ひずみ関係の評価

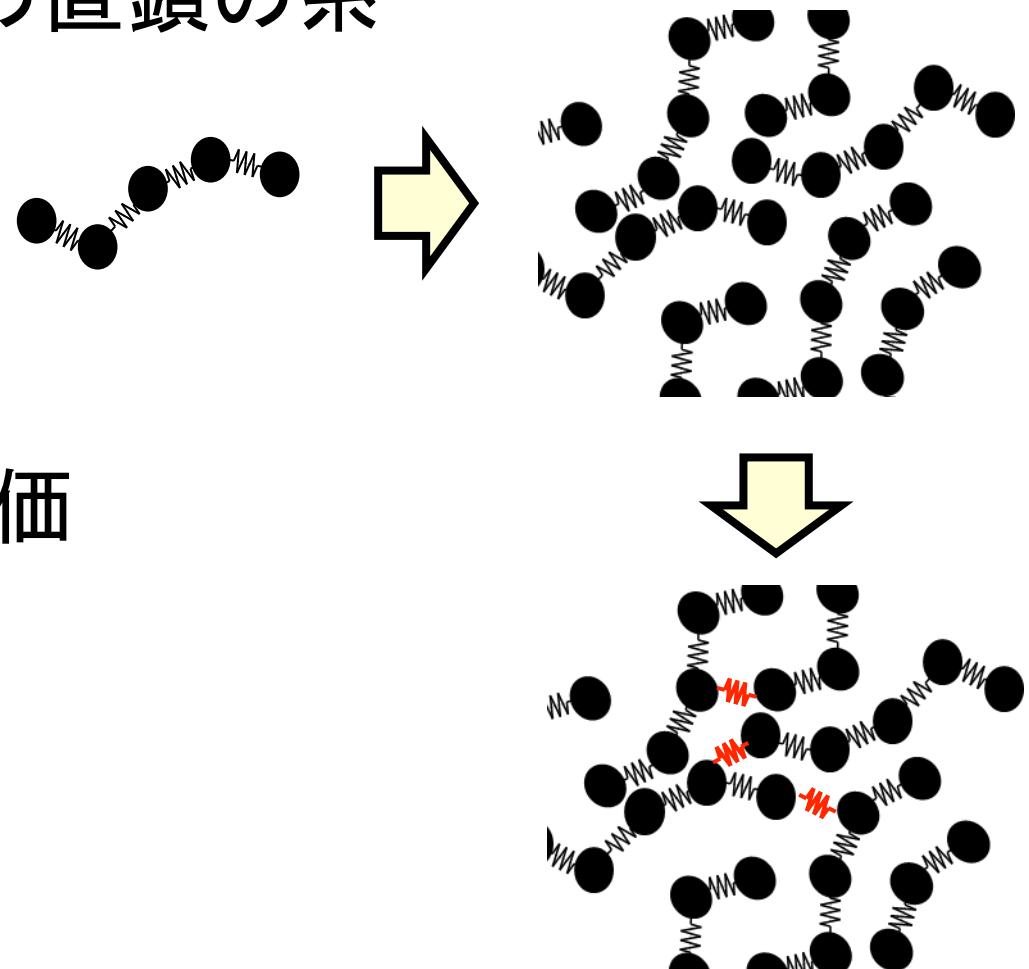
高度情報科学技術研究機構(RIST)
利用支援部 吉澤香奈子

Outline

- 架橋反応の実施(bond/create)
 - Kremer-Grestの1本鎖 (N=5) 【kantan1】
 - 多数本の鎖の系の作成 【kantan2】
 - 架橋の作成 【kantan3】
 - 変形と応力の計測 【kantan4】

架橋ポリマーの応力歪み関係

- バネビーズ模型の直鎖の系
 - 1本→多数本
 - 架橋生成
 - 変形+応力評価



Kremer-Grestの1本鎖 (N=5)(1)

データファイル

LAMMPS Description

5 atoms

4 bonds

1 atom types

1 bond types

-10.0 10.0 xlo xhi

-10.0 10.0 ylo yhi

-10.0 10.0 zlo zhi

Masses

1 1

Atoms

1 1 1 0.0 0.0 0.0

2 1 1 0.0 0.0 1.0

3 1 1 0.0 0.0 2.0

4 1 1 0.0 0.0 3.0

5 1 1 0.0 0.0 4.0

Bonds

1 1 1 2

2 1 2 3

3 1 3 4

4 1 4 5

データフォーマット

Masses

atom type番号, 質量

Atoms

粒子番号, Mol-ID, atom type, x, y, z, **wx**, **wy**, **wz**

周期境界条件
のwrap数

Velocities

粒子番号, vx, vy, vz

Bonds

ボンド番号, bond type, bond1, bond2

※ 速度データを含む場合
(AtomsとBondsの間)

Velocities
1 0.5 0.5 0.5

※ MassesとAtomsの間に、力場 パラメータを記録する
場合もある。

※ Bondsの後に、Angles, Dihedrals, Impropersを記録する
場合もある。

Kremer-Grestの1本鎖 (N=5)(2)

データファイル

data.kantan1

LAMMPS Description

5 atoms
4 bonds
2 atom types
1 bond types
2 extra bond per atom

-10.0 10.0 xlo xhi
-10.0 10.0 ylo yhi
-10.0 10.0 zlo zhi

Masses

1 1
2 1
3 1

Atoms

1 1 2 0.0 0.0 0.0
2 1 1 0.0 0.0 0.1
3 1 1 0.0 0.0 2.0
4 1 1 0.0 0.0 3.0
5 1 2 0.0 0.0 4.0

Bonds

1 1 1 2
2 1 2 3
3 1 3 4
4 1 4 5

入力ファイル

in-kantan1

データファイル
data.kantan1
を読み込む

units lj
atom_style bond
special_bonds fene

neighbor 0.4 bin
neigh_modify every 1 delay 1 check yes

Lennard-Jones(lj)
bead-spring polymers

隣接ペアリスト
作成の設定

read_data data.kantan1

velocity all create 0.0 12345 dist uniform

速度の設定

reset_timestep 0

開始の時刻(MDstep)の設定

pair_style lj/expand 2.5

pair_coeff * * 1.0 1.0 0.0 2.5

力場パラメータの設定

bond_style fene

bond_coeff 1 30.0 1.5 1.0 1.0

fix 1 all nve

NVEアンサンブル

fix 2 all langevin 1.0 1.0 2.0 48279

timestep 0.001

thermo_style multi

thermo 10000

時間刻みの設定

レポートの出力設定

restart

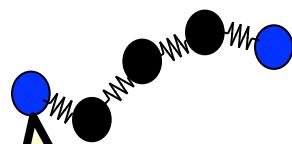
200000 restartB01

run

200000

restart-fileの出力設定

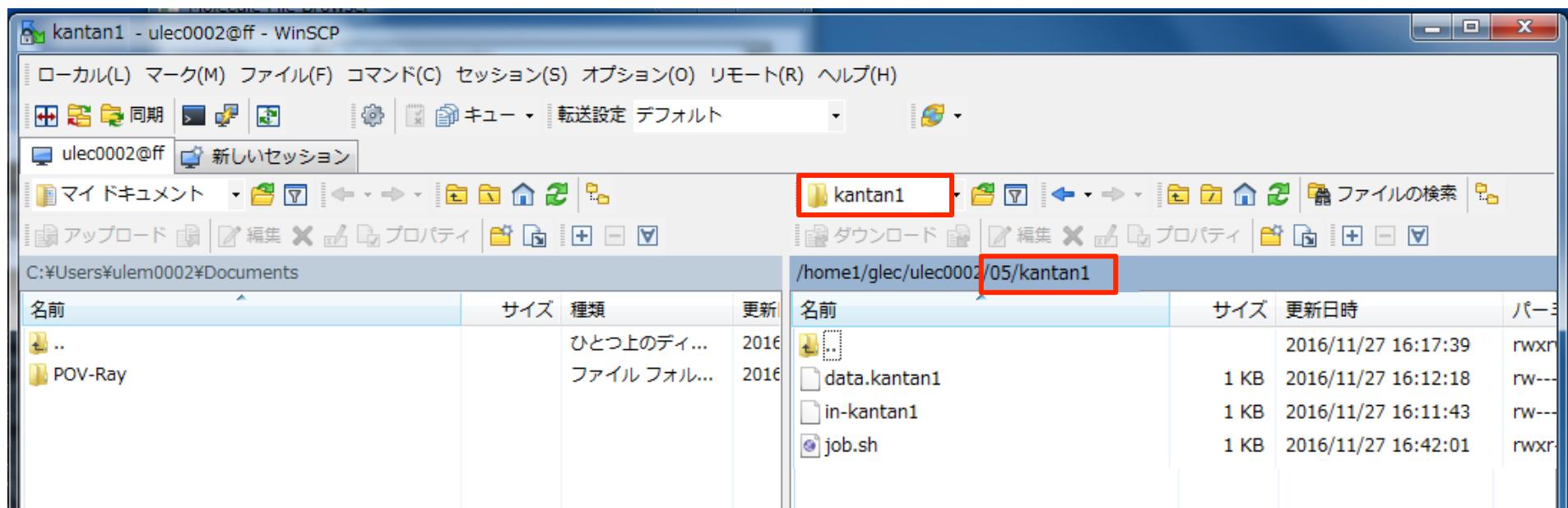
計算実行(step)数



両末端は、
あとで架橋
するように
Type=2

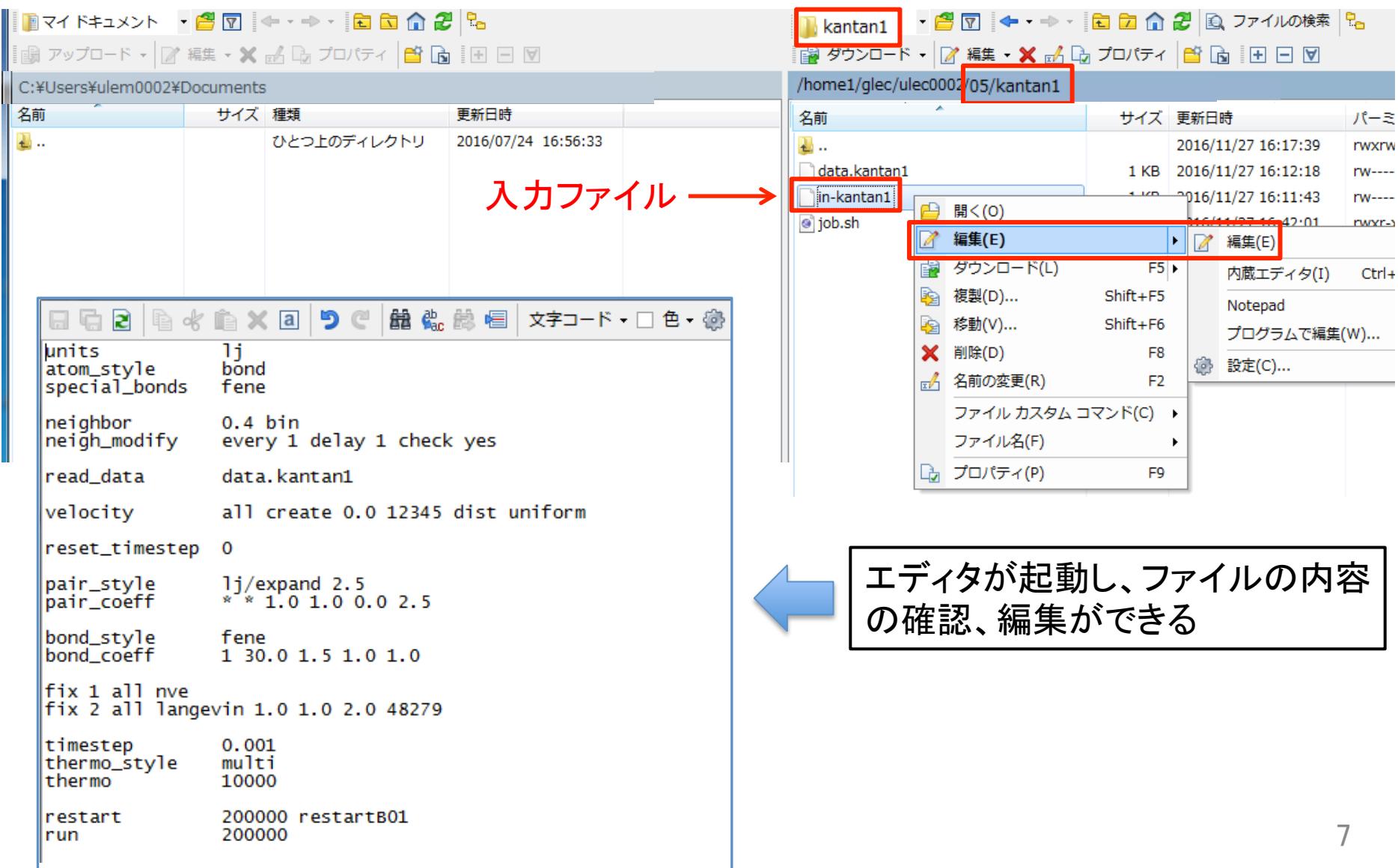
kantan1の実行(1)

- FOCUSスパコンへログイン
- WinSCPの起動
- \$HOME/05/kantan1 へ移動



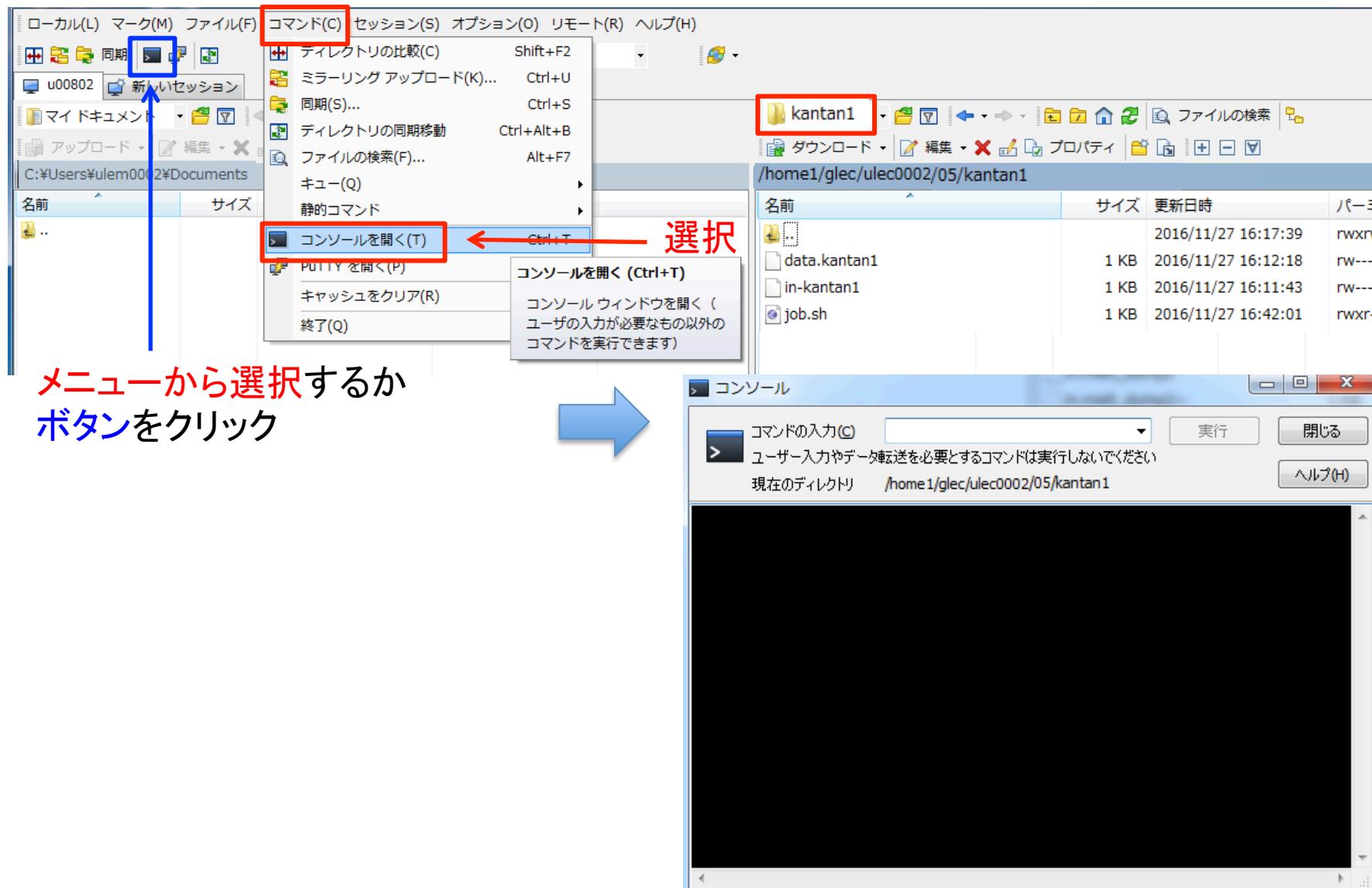
kantan1の実行(2)

- kantan1 の入力ファイル in-kantan1 を開く(中身の確認)



kantan1の実行(3)

- コンソールの起動



kantan1の実行(4)

- バッチスクリプト(job.sh)の確認

```
#!/bin/bash
#SBATCH -p e001h_lec
#SBATCH -n 2
#SBATCH -J Imp
#SBATCH -o Imp.o%J
#SBATCH -e Imp.e%J

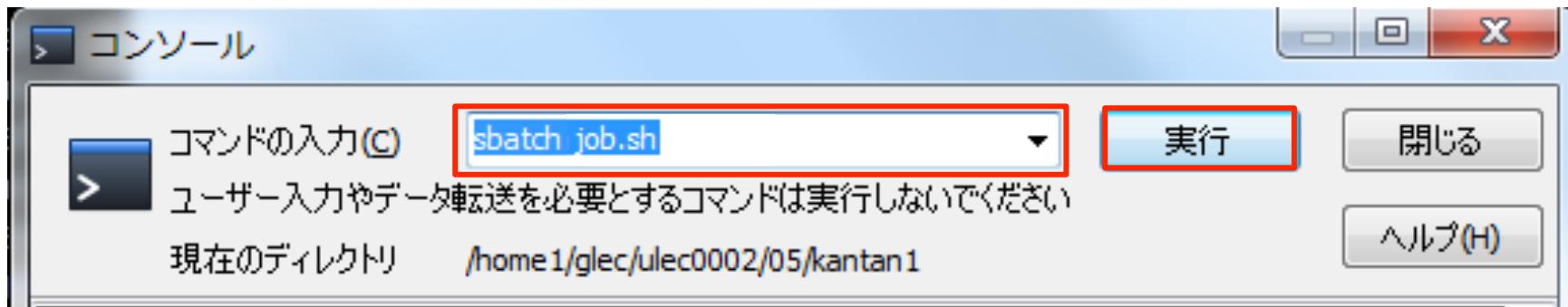
module load PrgEnv-intel
module load impi

export EXE=/home1/share/LAMMPS/lammps-30Jul16/src/Imp_mpi

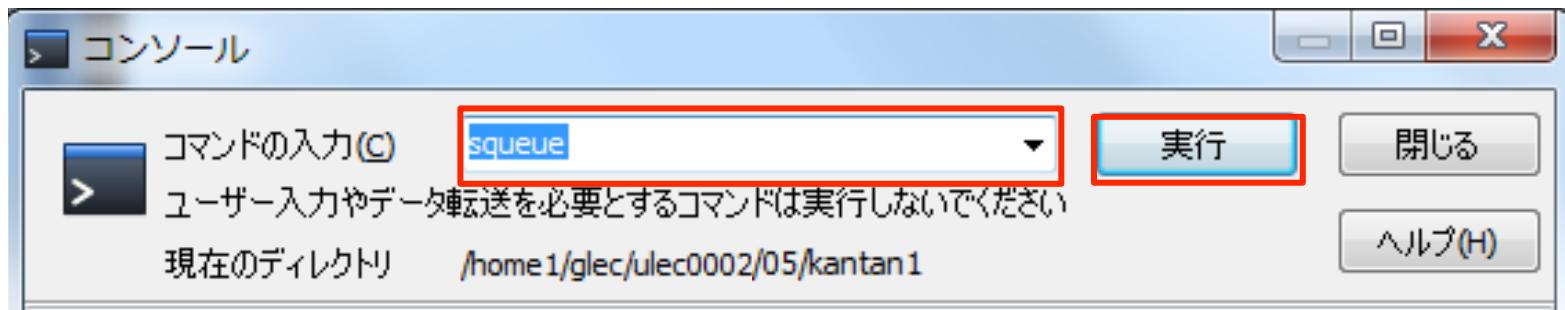
mpirun -np ${SLURM_NTASKS} $EXE < in-kantan1
```

kantan1の実行(5)

- \$ sbatch job.sh
を実行



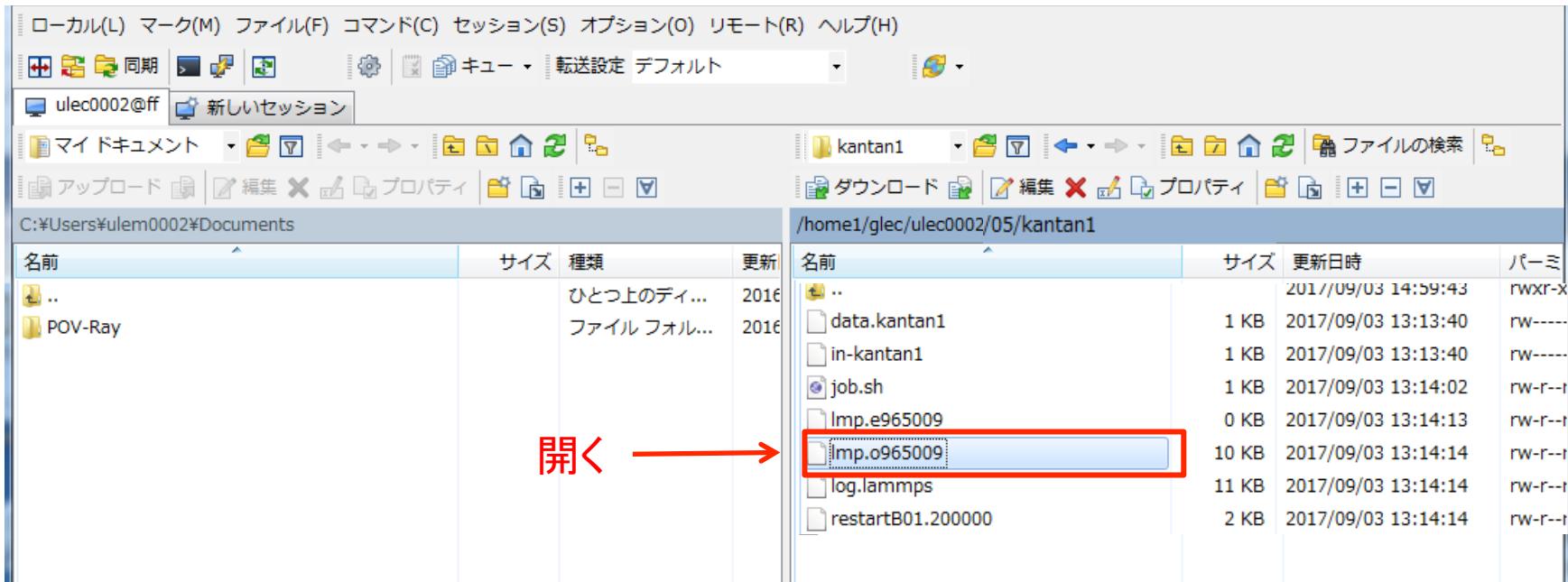
- 投入したジョブ状態の確認
\$ squeue



kanta1の実行(6)

- 実行結果の確認

```
$ less Imp.o*
```



kantan1の実行(7)

- 実行結果

```
E_coul = 0.0000 E_long = 0.0000 Press = 0.002/
----- Step 180000 ----- CPU = 0.3817 (sec)
TotEng = 17.7760 KinEng = 1.9723 Temp = 1.6436
PotEng = 15.8037 E_bond = 16.2567 E_angle = 0.0000
E_dihed = 0.0000 E_impro = 0.0000 E_vdw1 = -0.4530
E_coul = 0.0000 E_long = 0.0000 Press = -0.0014
----- Step 190000 ----- CPU = 0.4033 (sec)
TotEng = 17.4612 KinEng = 1.4901 Temp = 1.2418
PotEng = 15.9710 E_bond = 16.3154 E_angle = 0.0000
E_dihed = 0.0000 E_impro = 0.0000 E_vdw1 = -0.3444
E_coul = 0.0000 E_long = 0.0000 Press = 0.0003
----- Step 200000 ----- CPU = 0.4531 (sec)
TotEng = 17.6182 KinEng = 1.1143 Temp = 0.9286
PotEng = 16.5039 E_bond = 17.3064 E_angle = 0.0000
E_dihed = 0.0000 E_impro = 0.0000 E_vdw1 = -0.8025
E_coul = 0.0000 E_long = 0.0000 Press = -0.0008
Loop time of 0.453121 on 1 procs for 200000 steps with 5 atoms

Performance: 38135513.529 tau/day, 441383.258 timesteps/s
93.6% CPU use with 1 MPI tasks x no OpenMP threads

MPI task timing breakdown:
Section | min time | avg time | max time | %varavg| %total
-----+-----+-----+-----+-----+-----+
Pair | 0.095368 | 0.095368 | 0.095368 | 0.0 | 21.05
Bond | 0.053457 | 0.053457 | 0.053457 | 0.0 | 11.80
Neigh | 0.021548 | 0.021548 | 0.021548 | 0.0 | 4.76
Comm | 0.046104 | 0.046104 | 0.046104 | 0.0 | 10.17
Output | 0.028534 | 0.028534 | 0.028534 | 0.0 | 6.30
Modify | 0.15238 | 0.15238 | 0.15238 | 0.0 | 33.63
Other | 0.05573 | 0.05573 | 0.05573 | 0.0 | 12.30

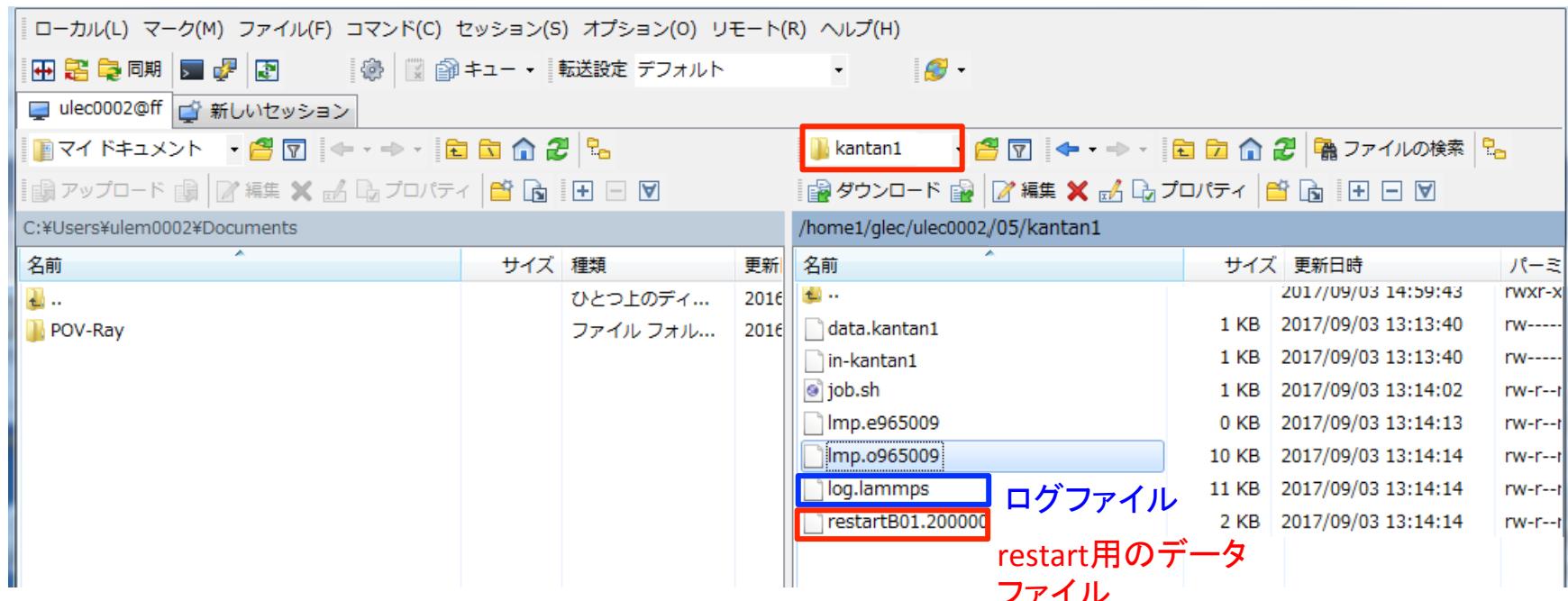
Nlocal: 5 ave 5 max 5 min
Histogram: 1 0 0 0 0 0 0 0 0
Nghost: 5 ave 5 max 5 min
Histogram: 1 0 0 0 0 0 0 0 0
Neighs: 6 ave 6 max 6 min
Histogram: 1 0 0 0 0 0 0 0 0

Total # of neighbors = 6
Ave neighs/atom = 1.2
Ave special neighs/atom = 1.6
Neighbor list builds = 2069
Dangerous builds = 0
Total wall time: 0:00:00
```

エラーメッセージなしで
Total wall time が表示されれば
正常終了

kantan1の実行(8)

- ログファイルと出力されたファイルの確認



多数本の鎖の系の作成

- restart_file からの、リストア

in-kantan2

系を複製
(8x8x8)

系のsize
を変形

```
units      lj
atom_style bond
special_bonds fene

neighbor   0.4 bin
neigh_modify every 1 delay 1 check yes
```

```
read_restart  restartB01.200000
reset_timestep 200000
```

```
replicate 8 8 8
```

```
pair_style    lj/expand 2.5
pair_coeff    * * 1.0 1.0 0.0 2.5
```

```
bond_style    fene
bond_coeff    1 30.0 1.5 1.0 1.0
```

```
fix 1 all nve
```

```
fix 2 all langevin 1.0 1.0 2.0 48279
```

```
fix 3 all deform 1 x final -7.22 7.22 y final -7.22 7.22 z final -7.22 7.22 remap x
```

```
timestep    0.001
```

```
thermo_style multi
```

```
thermo      10000
```

```
restart     200000 restartB02
```

```
run        200000
```

in-kantan1

data.kantan1

restartB01.200000

in-kantan2

...

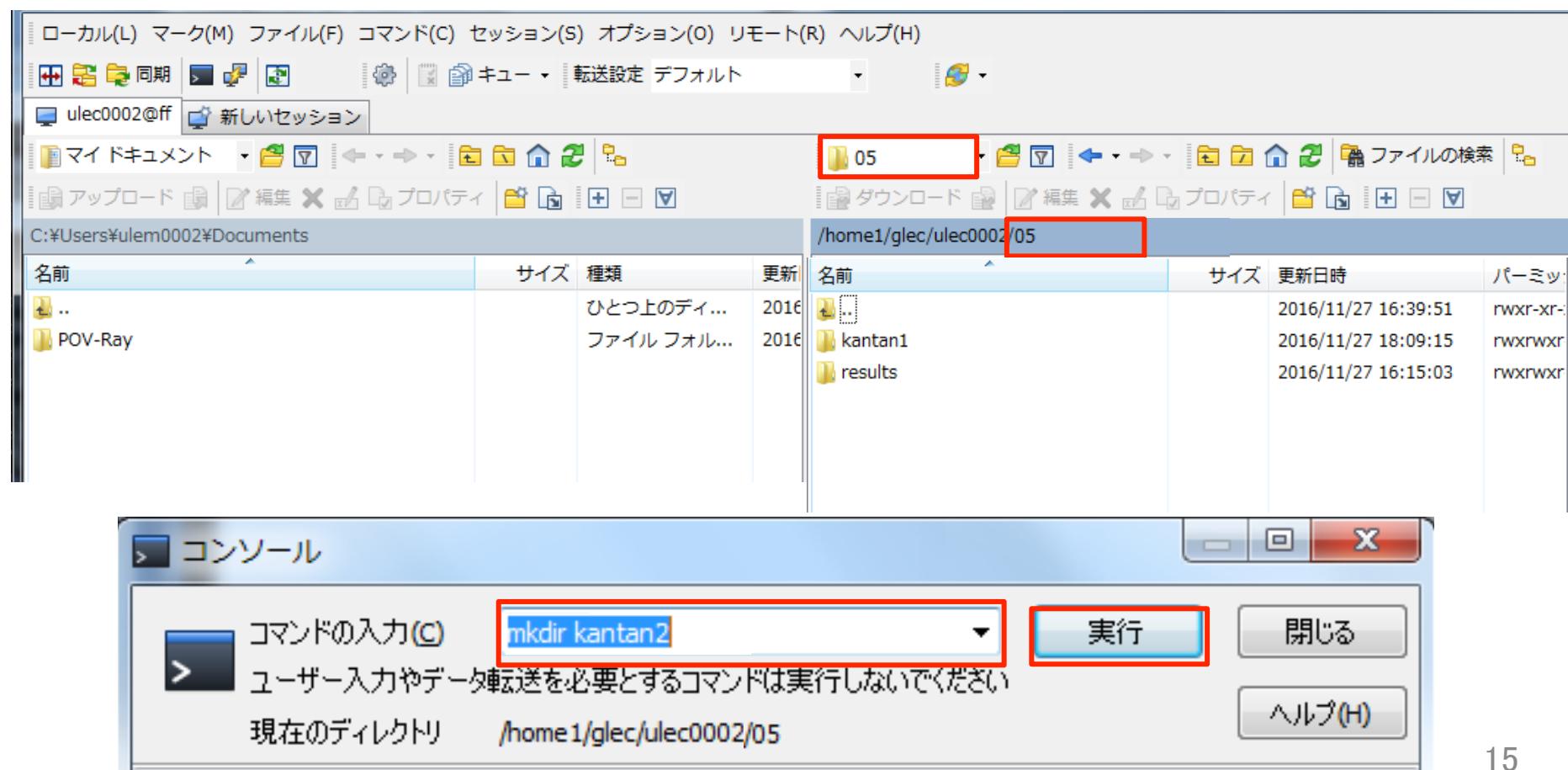
restartB02.400000

$$\left(\frac{5 \times 8^3}{0.85} \right)^{1/3} \approx 14.44$$

(80.0)³ から、(14.44)³ にスケール。

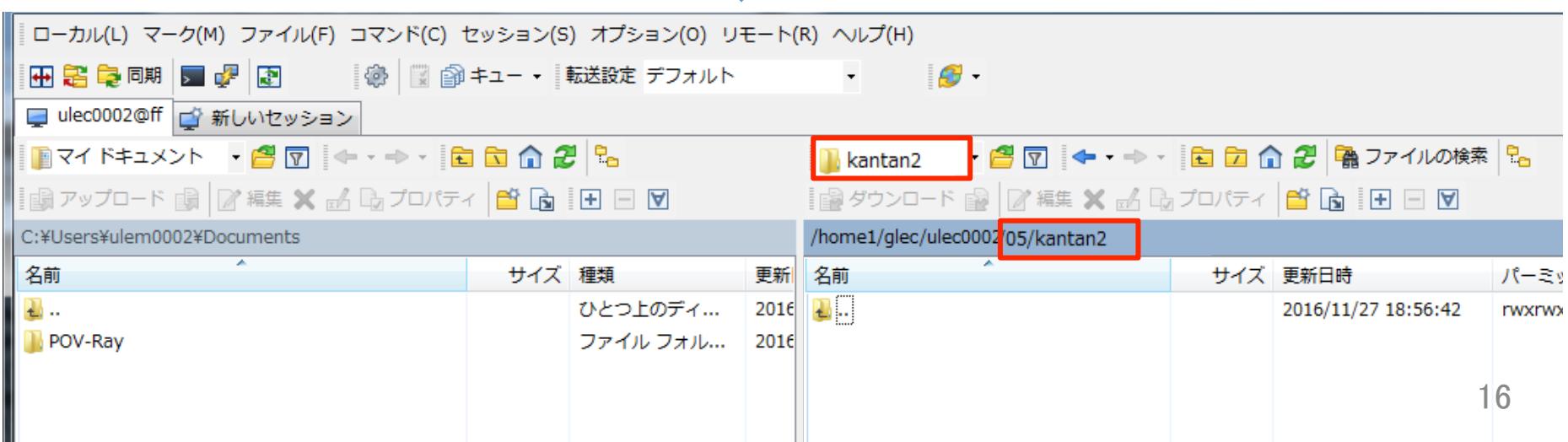
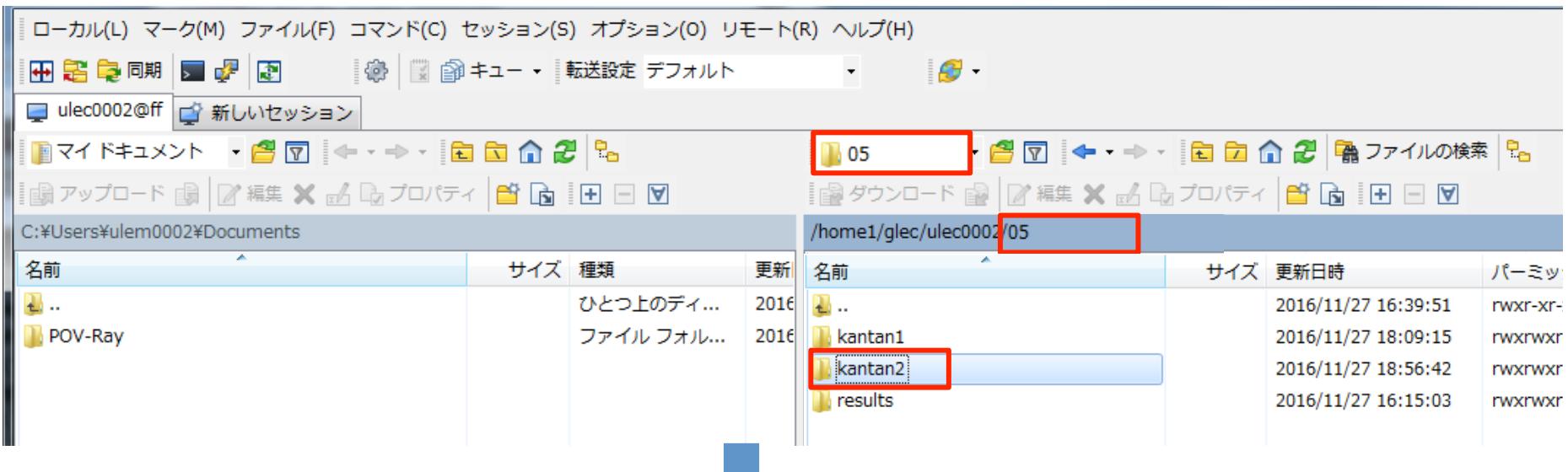
kantan2の実行(1)

- \$HOME/05 へ移動
- ディレクトリ kantan2 の作成
\$ mkdir kantan2



kantan2の実行(2)

- \$HOME/05/kantan2 へ移動



kantan2の実行(3)

- \$HOME/05/kantan1 からファイルのコピー
(in-kantan1, restartB01.200000, job.sh の3つのファイルをコピー)
\$ cp/kantan1/in-kantan1 in-kantan2 **in-kantan2 に名前を変えた**
\$ cp/kantan1/restartB01.200000 ./
\$ cp/kantan1/job.sh ./
- in-kantan2 の編集
P14 の in-kantan2 の内容と同じになるようにする。
- job.sh の編集
in-kantan1 → in-kantan2 へ変更
- \$ sbatch job.sh を実行
- ログファイル(正常終了)と出力されたファイルの確認
\$ ls
restartB02.400000 が出来ているか？

架橋の作成(1)

Fix-bond-createコマンド

```
fix 4 all bond/create 1 1 2 1.1 2 iparam 3 3  
jparam 2 3 prob 0.1 12345
```

作成するボンド種の最大数

作成するボンド種の最大に到達後に変更する原子種類

作成確率

作成するボンド種の最大数

チェックの頻度

原子種類 i

原子種類 j

判定距離

作成するボンド種

※ Fix-bond-createコマンドを使うにはMCパッケージを追加する必要がある。

架橋の作成(2)

- Extra の設定、newton on off の設定

in-kantan3

Newton 設定の変更

Fix-bond -create コマンド

```
units      lj
atom_style bond
special_bonds fene extra 16
newton on off

neighbor 0.4 bin
neigh_modify every 1 delay 1 check yes

read_data   rst.data
reset_timestep 400000

pair_style   lj/expand 2.5
pair_coeff   * * 1.0 1.0 0.0 2.5

bond_style   fene
bond_coeff   1 30.0 1.5 1.0 1.0

fix 1 all nve
fix 2 all langevin 1.0 1.0 2.0 48279
fix 4 all bond/create 1 1 2 1.1 1 iparam 3 3 jparam 2 3 prob 0.1 12345

timestep    0.001
thermo_style multi
thermo       10000

restart     2000000 restartB03
run         200000
```

Bond等のExtra情報
配列の大きさ

rst.dataのファイル中に、
16 extra bond per atom
を書き加えておく

2560 atoms
3 atom types
2048 bonds
1 bond types
16 extra bond per atom

Type1,2の粒子が、
距離1.1以下の場合、
確率0.1でbond作成

Bond作成後
Type=3に変更

kantan3の実行(1)

- \$HOME/05/kantan3 へ移動
- kantan3 に job_data.sh, job_check.sh ファイルがあることを確認
\$ ls
- \$HOME/05/kantan2 からファイルのコピー
(in-kantan2, restartB02.400000, job.sh の3つのファイルをコピー)
\$ cp/kantan2/in-kantan2 in-kantan3 in-kantan3 に名前を変えた
\$ cp/kantan2/restartB02.400000 ./
\$ cp/kantan2/job.sh ./
- in-kantan3 の編集
P19 の in-kantan3 の内容と同じになるようにする。
- job.sh の編集
in-kantan2 → in-kantan3 へ変更

kantan3の実行(2)

- データファイル rst.data の作成

Imp_mpi -r restartB02.400000 rst.data の実行

\$ sbatch job_data.sh

- Imp_data.o* ファイルの内容の確認

\$ less Imp_data.o*

LAMMPS (30 Jul 2016)

Reading restart file ...

restart file = 30 Jul 2016, LAMMPS = 30 Jul 2016

WARNING: Restart file used different # of processors (../read_restart.cpp:717)

orthogonal box = (-7.22 -7.22 -7.22) to (7.22 7.22 7.22)

1 by 1 by 1 MPI processor grid

2560 atoms

2048 bonds

Finding 1-2 1-3 1-4 neighbors ...

Special bond factors lj: 0 1 1

Special bond factors coul: 0 1 1

2 = max # of 1-2 neighbors

2 = max # of special neighbors

- rst.data ファイルが出来ているか確認

\$ ls

kantan3の実行(3)

- rst.data の編集

“1 bond types” の下に “**16 extra bond per atom**” を追加し、不要部分を削除。

LAMMPS data file via write_data, version 30 Jul 2016, timestep = 400000

2560 atoms

3 atom types

2048 bonds

1 bond types

16 extra bond per atom

追加

-7.21999999999998e+00 7.21999999999989e+00 xlo xhi

-7.21999999999998e+00 7.21999999999989e+00 ylo yhi

-7.21999999999998e+00 7.21999999999989e+00 zlo zhi

Masses

1 1

2 1

3 1

Pair Coeffs # lj/expand

1 1 1 0

2 1 1 0

3 1 1 0

削除

Bond Coeffs # fene

1 30 1.5 1 1

Atoms # bond

kantan3の実行(4)

- \$ sbatch job.sh
を実行
ログファイル(正常終了)と出力されたファイルの確認
- restartB03.600000 が出来ているか確認
\$ ls
- ボンド数の確認
Imp_mpi -r restartB03.600000 rst3.txt の実行
\$ sbatch job_check.sh
Imp_check.o* ファイルの内容の確認
\$ less Imp_check.o*
LAMMPS (30 Jul 2016)
Reading restart file ...
restart file = 30 Jul 2016, LAMMPS = 30 Jul 2016
WARNING: Restart file used different # of processors (../read_restart.cpp:717)
orthogonal box = (-7.22 -7.22 -7.22) to (7.22 7.22 7.22)
1 by 1 by 1 MPI processor grid
2560 atoms
3072 bonds

変形と応力の計測(1)

- Fix-deformコマンド
deform文での変形（体積保存）
`fix 5 all deform 10 x erate 0.005 y volume z volume
remap x units lattice`
- Compute文と、fix文での時間方向平均
`compute pres all pressure thermo_temp
fix 6 all ave/time 2000 1 2000 c_pres[1] c_pres[2]
c_pres[3] c_pres[4] c_pres[5] c_pres[6] file press.txt
ave one`

変形と応力の計測(2)

in-kantan4

```
units      lj
atom_style bond
special_bonds fene

neighbor 0.4 bin
neigh_modify every 1 delay 1 check yes
```

```
read_restart restartB03.600000
```

```
reset_timestep 0
```

タイムステップ
(時刻)を0にする。

Fix-deform
コマンド

```
pair_style lj/expand 2.5
pair_coeff * * 1.0 1.0 0.0 2.5
```

```
bond_style fene
bond_coeff 1 30.0 1.5 1.0 1.0
```

```
fix 1 all nve
fix 2 all langevin 1.0 1.0 2.0 48279
fix 5 all deform 10 x erate 0.005 y volume z volume remap x units lattice
```

```
compute pres all pressure thermo_temp
fix 6 all ave/time 2000 1 2000 c_pres[1] c_pres[2] c_pres[3] c_pres[4] c_pres[5] c_pres[6] file press.txt ave one
```

```
boxtimestep 0.001
thermo_style multi
thermo 10000
```

```
restart 200000 restartB04
run 200000
```

圧力の
出力

kantan4の実行(1)

- \$HOME/05 へ移動
- ディレクトリ kantan4 の作成

```
$ mkdir kantan4
```

- \$HOME/05/kantan4 へ移動

- \$HOME/05/kantan3 からファイルのコピー

(in-kantan3, restartB03.600000, job.sh の3つのファイルをコピー)

```
$ cp ..../kantan3/in-kantan3 in-kantan4
```

in-kantan4 に名前を変えた

```
$ cp ..../kantan3/restartB03.600000 ./
```

```
$ cp ..../kantan3/job.sh ./
```

- in-kantan4 の編集

P25 の in-kantan4 の内容と同じになるようにする。

- job.sh の編集

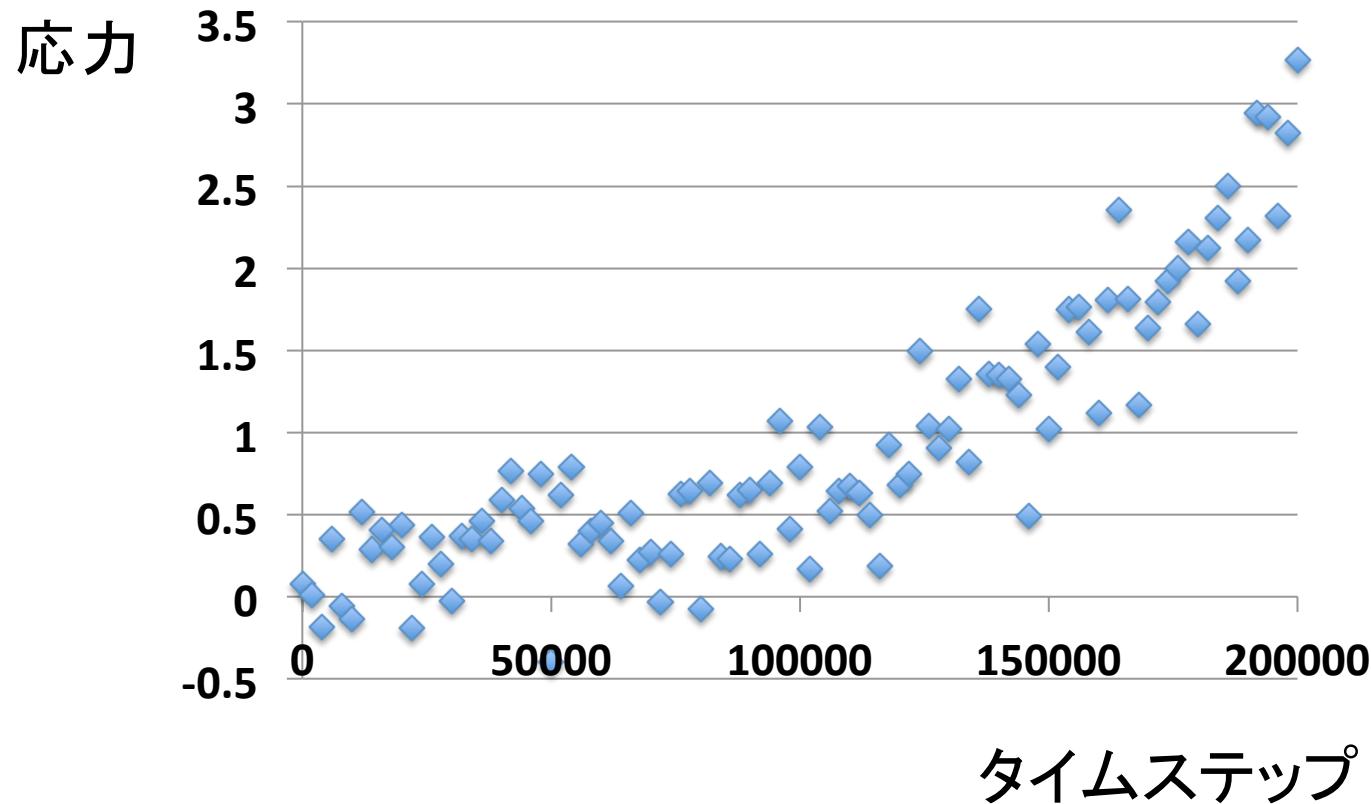
in-kantan3 → in-kantan4 へ変更

kantan4の実行(2)

- \$ sbatch job.sh
を実行
- ログファイル(正常終了)と出力されたファイルの確認
\$ ls
press.txt が出来ているか？
- 出力データの変換
\$ awk 'BEGIN {OFS=","} {print \$1,-\$2+(\$3+\$4)/2}' press.txt
> plot-data.csv

kantan4の実行(3)

- 結果のプロット
plot-data.csv をプロット



結果の確認

- \$HOME/05/results の中の
ディレクトリ kantan1 ~ kantan4
に入力ファイルと実行結果があるので、
上手くいかなかった人は確認して下さい。