

# MDSPASS (md2009) 日本語マニュアル

Y. Umeno

平成 22 年 10 月 28 日

## 1 Tutorial

1. パッケージをダウンロード, 展開する

```
tar xvfz md2008.tar.gz
cd md2008
```

2. 計算条件ファイルを指定

```
cp setdat.sample/SETDAT.typical SETDAT
```

3. コンフィグレーションファイルを指定

```
cp config.sample/CONFIG.Si333 CONFIG
```

4. 実行

```
./mdspass
```

5. データ出力

```
gnuplot
> p 'energy.d' title 'potential energy' w l,'u 1:4 title 'total energy' w l
```

\* ymd.binary が動かないときには,

```
./build
```

とすると, src ディレクトリでコンパイルされた実行ファイル (mdspass) がカレントディレクトリにコピーされるので, それを実行する.

\* Segmentation fault などでは実行できない場合には, src ディレクトリ内の Makefile を開き,

```
FC = ifort
```

を

```
FC = ifort -static
```

としてみる. (Warnings が出るが無視)

## 2 Input/output files

### 2.1 Input

- SETDAT: 計算条件が書き込まれたファイル.
- CONFIG: 原子配置, 原子種および使用するポテンシャルが書かれたファイル.
- CONT: SETDAT で `continue_md yes` となっているとき, ここから情報が読み込まれ計算が実行される.
- CONT.VAR: SETDAT で `continue_md yes` となっているとき, ここから FIRE アルゴリズムに関連する情報が読み込まれ計算が実行される.

### 2.2 Output

- CONFIG.END/CONFIG.END.FRA: CONFIG ファイルと同じであるが, 計算終了時のデータが書き込まれる. CONFIG.END.FRA は原子配置が fractional で書かれている.
- energy.d: ステップ毎のポテンシャルエネルギー, 運動エネルギー, 全エネルギー. eV/atom.
- stress.d: ステップ毎の応力成分 (xx,yy,zz,xy,xz,yx,yz,zx,zy). 単位は GPa.
- conf.xyz/conf.xsf/conf.cfg: 原子配置の xyz, xsf および cfg 形式出力. 10 ステップ毎に上書きされる. (Note: xyz は rasmol や xcrysden, xsf は xcrysden, cfg は atomeye などで見ることが可能. )
- cell.d: ステップ毎のセルサイズ (Å). h1x,h1y,h1z,h2x,h2y,h2z,h3x,h3y,h3z の順.
- LAST: 計算を終了した際の, 原子配置や速度などの情報が書かれている. これを CONT にコピーして (cp LAST CONT), `continue_md yes` で実行すれば計算が実行される.
- LAST.VAR: 計算を終了した際の, FIRE アルゴリズムに関連する情報が書かれている. これを CONT.VAR にコピーして (cp LAST.VAR CONT.VAR), `continue_md yes` で実行すれば計算が実行される.

### 2.3 Potential function

EAMPARAM.\* は EAM ポテンシャルを用いた計算のときに読み込まれる. (Please don't touch!)  
pot/ ディレクトリに置かれている必要がある (16.8.06 変更).

## 3 SETDAT file

SETDAT ファイル. コメントを見ればそれぞれの command の意味がわかる (はず). なお, Instability check については 7 章を参照.

```

#
# SETTING FILE
#
# For real number input, use double precision form
# unit should be in MKS (e.g. m, K, Pa)
# For logical input, not only .true./.false.
# but YES/Yes/yes/NO/No/no are acceptable.
#
# For more detail, please check 'readcondition.F'
#

# Time step 時間ステップ
dt 1.0d-15

# Setting cut-off radius.. not recommended カットオフ半径 (無視してください)
#rc 8.0d-10
# Setting cut-off radius for book-keeping. ブックキーピング半径
#frc 3.0d-10
# Interval of bookkeeping ブックキーピングを何ステップ毎に行うか
#nbk 20

# Iteration number MD のステップ数
#step_ite 100

# Tolerance (energy change(eV/atom) or max.force(eV/ang) or stress change (Pa)) MD の終了
# 条件を指定. テスト不十分のため怪しい.
# negative value or comment out to ignore
#tolfor 1.0e-3
# Energy tolerance
#tolene -1.0e-10
# Stress tolerance
#tolst -1.0e-3

# Velocity scaling (nonzero means interval steps) 速度スケーリング
#vscale 0

# Atom damper for relaxation (1=GLQC, 2=FIRE) 緩和計算用ダンパ
#damper 0

# Constraint of cell-deformation (works for cell-changing algorithm) セルの変形を拘束する
#cellconstx no
#cellconsty no
#cellconstz no

```

```

# Algorithm number アルゴリズム (アンサンブル) の指定
#      ! 0=Velocity Verlet, NVE (Allen & Tildesley)
#      ! 1=Gear's P-C, NVE (Allen & Tildesley)
#      ! 2=Gear, NVT (Nose thermostat)
#      ! 3=Gear, NPH (Andersen pressure control)
#      ! 4=Gear, NPT (Nose and Andersen)
#      ! 5=Gear, NPH-PR (Parrinello-Rahman)
#      ! 6=Gear, NPT-PR (Nose and Parrinello-Rahman)
#      ! 7=Stress relaxation by annealing, no atom motion
#      ! 8=Stress relaxation by annealing, atom motion by verlet
#      ! 50=Gear, Stress relaxation by NPH-PR & breaking
algorithm 0

# Weight of piston/thermostat (positive value) 圧力・温度制御の際のウェイト
# (1.0d0 is default. large value for slow response.)
stress_weight 1.0d0
temp_weight 1.0d-1

# Periodic boundary 周期境界条件
pbcx yes
pbcy yes
pbcz yes

# Target temperature 設定温度
temp_set 100.0d0

# Target stress 設定応力
# For Andersen method, use sgm_set
sgm_set 0.0d6
# For Parrinello-Rahman method, use following values
sxx 0.0d6
syy 0.0d6
szz 0.0d6
syx 0.0d6
syz 0.0d6
sxx 0.0d6
sxy 0.0d6

# Continue? If yes, CONT is used. (should be copied from LAST) 継続?
# Otherwise CONFIG is used. (Remember: last conf is saved to CONFIG.END)
continue_md no

# Whether initial velocity is given (does not work when contine_md) 初期温度を与える?

```

```

ini_velocity yes

# MD 計算をある一定時間行った後温度を操作したい時は、以下を指定する.
# appl_temp_step 温度操作を開始するステップ数
# appl_temp_length 温度操作を継続するステップ数
# appl_temp_temp1 その間、温度を何度にするかを設定
# appl_temp_temp2 温度操作終了時に、温度を何度に戻すかを設定
# 以下の例では、5000 ステップから 5010 ステップの間温度を 10K にし、5011 ステップで 0K にする.
appl_temp_step 5000
appl_temp_length 10
appl_temp_temp1 10.0
appl_temp_temp2 0.0

# Instability check
inst no

# 以下は無視してください (for experts)

# Infinitesimal strain for numerical evaluation of elastic constants
# If the value is positive, elastic constants are calculated at the end of MD
# Cubic symmetry is assumed
econstd -1.0d-3

# Voronoi polyhedron analysis: radius (ang), zero or negative means no analysis
voronoi -6.0

```

## 4 CONFIG file

### 4.1 Copper unit cell

CONFIG ファイル

```

Cu          <-- 原子種. 原子種が一つしかない場合はここに書く.
EMT_JSN     <-- ポテンシャル. この場合は EMT(Surface Science 366 (1996) 394-402).
      4     <-- 原子の数 N.
3.58736     <-- 格子定数 a(Å).
1.0 0.0 0.0 <-- 格子ベクトル 1. a 倍される.
0.0 1.0 0.0 <-- 格子ベクトル 2. a 倍される.
0.0 0.0 1.0 <-- 格子ベクトル 3. a 倍される.
fra         <-- 以下の原子配置がセルに対する相対座標で与えられることを示す. abs なら Å.
0.0 0.0 0.0 <-- 原子座標. N 行続く.
0.0 0.5 0.5

```

```
0.5 0.0 0.5
0.5 0.5 0.0
```

特定の原子を動かしたくない場合は，原子座標の各行に，x,y,z 座標に続き T(動かす) または F(固定) を書くことができる．たとえば，

```
0.0 0.0 0.0 F F F
0.0 0.5 0.5 T T F
0.5 0.0 0.5
0.5 0.5 0.0
```

は，「1 番目の原子は固定し，2 番目の原子は x, y 方向への動きだけを許し，3, 4 番目の原子は拘束しない」ことを示す．

現在使用可能なポテンシャルキーワードは Table 1 のとおり．

[注意書き]

- Tersoff には 2 種類のアルゴリズムが用意されている; (a) ボンドオーダー項 ( $b_{ij}$ ) を matrix として先に計算しておく, (b)  $b_{ij}$  が必要となる毎に逐一計算する．キーワードで “Tersoff” はアルゴリズム (a), “TersoffNM” はアルゴリズム (b) を指定する．前者は高速であるが小さなセルではエラーとなる．その場合には TersoffNM を試されたい．
- (追記 15.8.06) また，(a) はセルが大きい場合にもメモリ消費量が大きくなり計算不能となる (マシンにもよるが原子数 2000 程度以上は推奨できない)．その場合にも TersoffNM を試されたい．

## 4.2 SiC (binary system)

複数原子種がある場合は CONFIG ファイルは以下ようになる．すなわち，一行目で Multi と指定し，原子座標 x,y,z の右側に原子種を書く．

```
Multi
Tersoff M
      8
4.37
1.0000000000000000 0.0000000000000000 0.0000000000000000
0.0000000000000000 1.0000000000000000 0.0000000000000000
0.0000000000000000 0.0000000000000000 1.0000000000000000
fra
0.0000000000000000 0.0000000000000000 0.0000000000000000 Si
0.5000000000000000 0.5000000000000000 0.0000000000000000 Si
0.0000000000000000 0.5000000000000000 0.5000000000000000 Si
0.5000000000000000 0.0000000000000000 0.5000000000000000 Si
0.2500000000000000 0.2500000000000000 0.2500000000000000 C
0.7500000000000000 0.7500000000000000 0.2500000000000000 C
0.2500000000000000 0.7500000000000000 0.7500000000000000 C
0.7500000000000000 0.2500000000000000 0.7500000000000000 C
```

Table 1 Keywords for potential selection.

Keyword	Potential	Reference
EMT_ST1	EMT for Au	J.Chem.Phys. 92 (1990) p.6306
EMT_HM1	EMT for Cu	J.Phys.: Condens. Matter 1 (1989) p.9765
EMT_JSN	EMT for Cu, Ag, Au, Ni, Pd, Pt	Surface Science 366 (1996) p.394
EAM_MIS	Mishin potential for Al, Cu, Ni	PRB 59 (1999) p.3393
EAM Mishin	(Synonym of above)	
EAM_FS1	Finnis-Sinclair potential for Fe	Phil. Mag. A 50 (1984) p.45
EAM NiAl	EAM for NiAl by Mishin	PRB 65 (2002) art.224114
GEAM PRB	Generalized EAM for 16 species of metal	PRB 69 (2004) art.144113 <sup>*1</sup>
GEAM AM	Generalized EAM for 16 species of metal	Acta Mater. 49 (2001) p.4005
EAM GEAM	(Synonym of above)	
Tersoff B	Tersoff potential (B, surface structure)	PRB 38-14(1988)p.9902
Tersoff C	Tersoff potential (C, elastic property)	PRB 38-14(1988)p.9902
Tersoff B2	The same as B except for cutoff ( $R = 2.75, D = 0.1$ )	PRB 37-12(1988)p.6991
Tersoff M	Tersoff multisystem (Si-C, Si-Ge)	PRB 39-8(1989)p.5566
TersoffNM	Tersoff for small cells (see caveat in the text)	
StilWeb	Stillinger-Weber potential for Si	PRB 31, p.5262 (1985)
Shellmodel PT0	Shell Model for PbTiO <sub>3</sub>	
Morse	Morse potential	Phys. Rev. 114 (1959) p.687
MEAM SiC	Modified EAM (MEAM) for SiC by Huang et al.	MSMSE 3, pp.615-627 (1995)
MEAM Sn	Modified EAM (MEAM) for Sn by Ravelo et al.	PRB 79, pp.2482-2485 (1997)

<sup>\*1</sup> Gives the strange lattice constant for Al bulk.

一つめの原子だけを拘束したい場合

```
0.0000000000000000 0.0000000000000000 0.0000000000000000 Si F F F
```

と書くとい。

## 5 Modifications

### 5.1 領域分割法による Bookkeeping の効率化 (15.8.06)

Bookkeeping の際、領域分割法を援用することでスピードアップが可能となった。大規模系に適する。SETDAT ファイル内に

```
# Region partitioning (for accelerating bookkeeping)
rpart yes ! enable region partitioning
npcx 10 ! # of partition in x axis
npcy 10 ! # of partition in y axis
npcz 3 ! # of partition in z axis
```

のように記述する。npcx, npczy, npczy は大きい程効率化が見込めるが、各 partition の一辺が frc より小さくなることはできない。なお、orthorhombic な simulation cell にしか対応しておらず、かつセル形状ベクトルは x 軸, y 軸, z 軸 に (この順番で) 沿うようにセットしなければならない。

## 5.2 ブックキーピング・リスト テーブルサイズの設定 (15.8.06)

ブックキーピングに使用するリストテーブルサイズを事前に設定できるようにした。デフォルト値 (1000000) で足りない場合にはプログラム実行中に自動的に必要なサイズが再計算されるが、これには時間がかかるため、あらかじめ設定しておくのがよい。SETDAT 内に

```
nbook 2000000
```

のように記述する。

## 5.3 ディレクトリ構造の整理 (16.8.06)

ソースファイルは src/ 以下に置いた。ここでコンパイルし得られたバイナリファイル (ymd or ymd.binary) を working directory に移し、そこで計算を実行されたい。なお、EAM ポテンシャルファイル (EAMPARAM.\*) および MEAM リファレンスデータファイル (meam\_\*\_ref.d) は (working directory)/pot/ に置かれている必要がある。

## 5.4 セルの multiplication

ユニットセルなど、サイズの小さいセルを用いた場合、ポテンシャルのルーチンによってはセルサイズがカットオフ半径に比べて小さすぎるため計算できないことがある。しかしそのためにわざわざ CONFIG ファイルを書き換え、大きなシミュレーションセルを作るのが面倒な場合がある。そこで、与えられた CONFIG ファイルの原子配置・セルサイズを読み込んだあと、それを  $x, y, z$  方向にコピーして大きなセルを作成することで、この問題を回避することができる。例えば、SETDAT 内に

```
multicellx 3  
multicelly 3  
multicellz 2
```

と書くと、 $x, y, z$  方向にそれぞれ 3, 3, 2 倍したセル (原子数 18 倍) の計算となる。

## 6 Source of errors

本プログラムはあらゆる条件に対してテストされている訳ではないので、プログラムが robust でない。そのためシミュレーションの際には基本的なチェック (MD 計算中のエネルギーの挙動など) をした上で使用されたい。以下に開発中に明かとなった注意点を示す。

1. セルはできるだけ各辺が  $x, y, z$  軸に沿うように配置した方がよい (ブックキーピングのアルゴリズムが貧弱な為)。エネルギーのジャンプなどが nbk 毎に起こる場合はこの可能性が高い。やむを得ない場合は frc を高く設定することで回避可能であるがその分計算時間がかかる。



2. FIRE アルゴリズムを使用する場合, `continue_md = yes` による中断・再開をすると, 中断せず MD を継続した場合と結果が異なる. FIRE アルゴリズムで変化する慣性パラメータが引き継がれないためである. (要修正事項)

## 7 不安定性解析

本章では, Instability check の説明を行う.

### 7.1 基本: 固有値・固有ベクトル解析

SETDAT において `inst yes` とすれば, MD 計算終了時 (終了条件を満たしたとき) に Hessian マトリクスの固有値解析を行う. 固有値の小さいものから  $i = 1, 2, 3, \dots$  とする.

- `emin.d`: 小さいものから順に 10 個の固有値が (1 行に) 書き出される (`step e_1 e_2 ... e_10`).
- `eminall.d`: 小さいものから, 全ての固有値が書き出される (`i e_i`).
- `atominstXX.xsf`: 固有ベクトルが `xsf` 形式で書き出される (`XX` は 01~10).

`emin_cell.d` にはセルのひずみに対する不安定性解析が行われた結果が出力される (今のところ不使用).

### 7.2 固有ベクトル方向へのエネルギーカーブ・曲面解析

固有ベクトル  $\mathbf{p}_i$  (のスカラー倍) を変位として与え, エネルギーを計算する.

#### 7.2.1 単独モード方向

`inst_curve yes` とすれば単独モード方向へのエネルギーカーブが計算され ( $i = 1 \sim 10$ ) る. すなわち,

$$E(x) = E(\mathbf{R}_0 + x \cdot \mathbf{p}_i)$$

を求める. 結果は `ecurvXX.d` に出力される (`XX` は 01~10,  $x$  [m]  $E$  [J] の 2 コラムデータ).

#### 7.2.2 混合モード方向

`inst_surf yes` とすれば,

$$E(x, y) = E(\mathbf{R}_0 + x \cdot \mathbf{p}_i + y \cdot \mathbf{p}_j)$$

を求めることができる. 以下に例を示す.

```
# Energy surface calculation (two modes)
inst_surf yes
inst_surf_x 4  <-- i を指定
inst_surf_y 5  <-- j を指定
# range (in ang)
inst_surf_xmn -1.2  <-- x の範囲 (ここから), 単位はオングストローム.
```

```

inst_surf_xmx 1.2 <-- x の範囲 (ここまで)
inst_surf_ymn -1.2 <-- y の範囲 (ここから)
inst_surf_ymx 0.4 <-- y の範囲 (ここまで)

```

このとき, esurface-004-005.dにエネルギー曲面データが出力される ( $x$  [Å],  $y$  [Å],  $E$  [J]). また, 与えた変位は conf-esurf-XXXX-YYYY.cfg, 変位ベクトルは disp-esurf-XXXX-YYYY.xsf に出力される. XXXX, YYYY はそれぞれ  $x$  [Å]/0.04,  $y$  [Å]/0.04 であり, 正, 負を一桁目 (p/n) で表す. 例えば, -30 なら n030, 10 なら p010 となる.

### 7.3 ゆらぎ解析

あらかじめ有限温度 MD 解析をし, confXXX.cfge に原子配置のスナップショットを書き出しておく. その配置と 0K の最安定原子配置との差分をとり, Hessian マトリクスの固有ベクトル (モード) 成分に分解してその係数を求めることができる. 例えば,

```

# Instability analysis
inst yes
inst_curve no
# Instability analysis -- decomposition into eivenvectors
inst_decom yes
inst_decom_nfile 100

```

とすると, conf001.cfge から conf100.cfge までを解析する. 各モード成分の係数  $\alpha_i$  は, decomposeXXX.d に書き出される ( $i, \alpha_i$  の 2 コラムデータ). dispXXX.xsf に変位 (最安定配置との差分) が出力される.

† スナップショットの数だけ decomposeXXX.d が生成されるが, 各係数の時間に対する変動を見たい場合には, tools/ディレクトリにある alpha\_anal.F を使用すれば,  $\alpha_i$  の変動が alphatrajYYY.d に書き出される (スナップショット番号,  $\alpha_i$  の 2 コラムデータ). alpha\_anal.F 内の mxfile=XXX でスナップショット番号の最大値を指定しておく (上記の例の場合, 100).

さらに,

```

# Instability analysis
inst yes
inst_curve no
# Instability analysis -- decomposition into eivenvectors
inst_decom yes
inst_decom_nfile 100
inst_decom_curve yes
inst_decom_maxmode 10

```

とすると, 変位ベクトルからあるモード以上の成分を取り除いたベクトル (あるモードまでの成分の総和) を生成して, その方向へのエネルギーカーブを計算する. 上記の例では,  $i=1$  のみ,  $i=2$  まで,  $i=3$  まで, ...,  $i=10$  までのモード成分の総和をとり, 出来たベクトルのスカラー倍の変位を与えエネルギーを求める. 計算時間は長くなるので注意すること. 与えた変位ベクトルは dispXXX-uptoYYY.xsf に, エネルギーカーブは ecurveXXX-uptoYYY.d に書き出される. YYY が「何番目までのモード成分を足し合わせたか」を示す.

## 8 拡散解析

SETDAT において `diffusion yes` とすると、拡散解析モードとなる。拡散解析モードでは、初期状態からの原子の変位を MD 実行中に記憶しておくようになっている。msd.d というファイルが生成され、この変位から求めた平均自乗変位が書き出される。また、計算終了時には LAST.DIF ファイルが生成される。continue\_md yes で継続する場合にはこれを CONT.DIF という名前に書き換えておく必要がある。

MD 実行中に原子の変位データを書き出すことができる (これを元に原子の軌跡などが解析できる)。displacementXXX.d というファイルに保存される (XXX には数字が入る)。difout\_int は、何ステップ毎に変位データを書き出すかを指示する。displacementXXX.d には、原子番号  $i = 1$  から順に 1 行ごとに、

```
dis_x dis_y dis_z
```

が書き出される (単位はÅ)。

## 9 Shell Model による MD 計算

### 9.1 CONFIG ファイル

以下に、PbTiO<sub>3</sub> 単結晶の cubic ユニットセルの場合の例を示す。

```
Multi
Shellmodel PTO
      10
      3.8910
      1.0000000000000000 0.0000000000000000 0.0000000000000000
      0.0000000000000000 1.0000000000000000 0.0000000000000000
      0.0000000000000000 0.0000000000000000 1.0000000000000000
fra
      0.0000000000000000 0.0000000000000000 0.0000000000000000 Pb
      0.5000000000000000 0.5000000000000000 0.5000000000000000 Ti
      0.5000000000000000 0.5000000000000000 0.0000000000000000 O
      0.5000000000000000 0.0000000000000000 0.5000000000000000 O
      0.0000000000000000 0.5000000000000000 0.5000000000000000 O
      0.0000000000000000 0.0000000000000000 0.0000000000000000 Pb
      0.5000000000000000 0.5000000000000000 0.5000000000000000 Ti
      0.5000000000000000 0.5000000000000000 0.0000000000000000 O
      0.5000000000000000 0.0000000000000000 0.5000000000000000 O
      0.0000000000000000 0.5000000000000000 0.5000000000000000 O
```

Core と Shell の両方を擬似的な「原子」として扱うため、原子数は実際の 2 倍として与える。上記の例の場合、ユニットセルに含まれる原子数は実際は 5 であるが、3 行目には 10 を与える。fra の次行以下は、まず Core の配置を与え (5 行)、その後 Shell の配置 (5 行) を与える。

### 9.2 その他セッティング

Shell Model では、厳密には質量を持たない Shell が 質量を持つ Core の動きに即座に応答し最適な位置に収束するようにしなければならないが、コードを簡単化するため Shell にも質量を与え Shell と Core の両

方をあたかも異なる原子であるかのように扱い、同時に MD 計算する方法をとっている。このため、時間ステップを通常より 1 桁程度小さく設定する必要がある。

† 本当は、Shell の仮想質量を調整する必要があるが、現在は adjustable となっていない。これについては近々修正の予定。

## 10 実行例・サンプル シェルスクリプト

### 10.1 平衡格子定数を求める

理想結晶の原子配置ファイルを用意し、CONFIG.TMP にコピーしておく。  
SETDAT の主要なキーワードを以下のようにしておく。

```
frc 3.0d-10
step_ite 1
pbcx yes
pbcy yes
pbcz yes
algorithm 0
temp_set 0.0d0
ini_velocity no
```

下記のようなシェルスクリプトを実行すれば SUMMARY に計算結果が書かれる。

```
#!/usr/bin/perl
open (S, ">>SUMMARY");
$a=3.00;
$ainc=0.1;
while ($a<=4.50) {

    open(C, "<CONFIG.TMP");
    open(F, ">CONFIG");
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;print F "$a\n";
    while ($line=<C>) {
        chomp($line);
        print F "$line\n";
    }
    close(C);
    close(F);
    system("./ymd");

    # Extract energy output (last line) from energy.d
    open (T,"<energy.d");
```

```

while ($line=<T>) {
chomp($line);
@splitline=split(/\s+/, $line);
$result[1]= "$splitline[1]";
$result[2]= "$splitline[2]";
$result[3]= "$splitline[3]";
$result[4]= "$splitline[4]";
}
close (T);
$ene=$result[2];
print S "$a $ene\n";

$a=$a+$ainc;
}
close(S);

```

## 10.2 一方向ひずみ負荷 (Silicon 圧縮の例)

### 10.2.1 Energy landscape

CONFIG.TMP (ymd の Tersoff サブルーチンは小さなセルに対応していないので, 大きめのセルを準備)

```

Si
Tersoff B2
      64
      5.4310
      1.41421356 0.0 0.0
      0.0 1.41421356 0.0
      0.0 0.0 4.0
fra
0.0000000000000000 0.0000000000000000 0
0.2500000000000000 0.2500000000000000 0.125
0.2500000000000000 0.0000000000000000 0.0625
0.0000000000000000 0.2500000000000000 0.1875
...
(以下略)

SETDAT
dt 1.0d-15
frc 3.5d-10
nbk 20
step_ite 1
algorithm 0
pbcx yes

```

```

pbcy yes
pbcz yes
temp_set 0.0d0
continue_md no
ini_velocity no
シェルスクリプト

#!/usr/bin/perl
open (S, ">SUMMARY2");
$lz=1.00;          # lz の範囲を指定
$lzinc=-0.01;      #
while ($lz>=0.6) {  #
    $lx=1.00;       # lx の範囲を指定
    $lxinc=0.01;    #
    while ($lx<1.2) { #
        $lzz=$lz*2.0;
    open(C, "<CONFIG.TMP");
    open(F, ">CONFIG");
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;print F " $lx $lx 0.0\n";
    $line=<C>;print F " -$lx $lx 0.0\n";
    $line=<C>;print F " 0.0 0.0 $lzz\n";
    while ($line=<C>) {
        chomp($line);
        print F "$line\n";
    }
    close(C);
    close(F);
    system("./ymd");
# Extract energy output (last line) from energy.d
open (T,"<energy.d");
while ($line=<T>) {
    chomp($line);
    @splitline=split(/\s+/, $line);
    @result[1]= "$splitline[1]";
    @result[2]= "$splitline[2]";
    @result[3]= "$splitline[3]";
    @result[4]= "$splitline[4]";
}
close (T);
$ene=$result[2];

```

```

# Extract stress output (last line) from stress.d
open (T,"<stress.d");
while ($line=<T>) {
chomp($line);
@splitline=split(/\s+/, $line);
@result[1]= "$splitline[1]";
@result[2]= "$splitline[2]";
@result[3]= "$splitline[3]";
@result[4]= "$splitline[4]";
}
close (T);
$sxx=$result[2];$syy=$result[3];$szz=$result[4];
print S "$lx $lz $ene $sxx $syy $szz\n";
    $lx=$lx+$lxinc;
}
print S "\n";
$lz=$lz+$lzinc;
}
close(S);

```

## 10.2.2 $z$ 方向圧縮, $x, y$ 方向応力制御

```

dt 1.0d-15
frc 3.5d-10
nbk 20
step_ite 1000
tolst 1.0e-3      # Stress の変化量が小さくなれば終了
cellconstx no
cellconsty no
cellconstz yes    # z 方向にはセルサイズコントロールしない
algorithm 7
stress_weight 10.0d-1
pbcx yes
pbcy yes
pbcz yes
temp_set 0.0d0
# Set transverse stress here: 横方向応力設定
sxx 5000.0d6
syy 5000.0d6
continue_md no
ini_velocity no

```

シェルスクリプト

```

#!/usr/bin/perl
open (S, ">SUMMARY");
$lz=0.45;
$lzinc=-0.01;
while ($lz>=0.4) {

    $lzz=$lz*4.0;
    $ba1x=1.0;
    $ba1y=0.0;
    $ba1z=0.0;
    $ba2x=0.0;
    $ba2y=1.0;
    $ba2z=0.0;
    $ba3x=0.0;
    $ba3y=0.0;
    $ba3z=1.0;

    open(C, "<CONFIG.TMP");
    open(F, ">CONFIG");
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;chomp($line);print F "$line\n";
    $line=<C>;print F " 0.0 0.0 $lzz\n";
    while ($line=<C>) {
        chomp($line);
        print F "$line\n";
    }
    close(C);
    close(F);
    system("./ymd");

# Extract energy output (last line) from energy.d
open (T,"<energy.d");
while ($line=<T>) {
    chomp($line);
    @splitline=split(/\s+/, $line);
    @result[1]= "$splitline[1]";
    @result[2]= "$splitline[2]";
    @result[3]= "$splitline[3]";
    @result[4]= "$splitline[4]";
}

```



```

}
close (T);
$ene=$result[2];
# Extract stress output (last line) from stress.d
open (T,"<stress.d");
while ($line=<T>) {
  chomp($line);
  @splitline=split(/\s+/, $line);
  @result[1]= "$splitline[1]";
  @result[2]= "$splitline[2]";
  @result[3]= "$splitline[3]";
  @result[4]= "$splitline[4]";
}
close (T);
$sxx=$result[2];$syy=$result[3];$szz=$result[4];

# Extract cell-size output (last line) from cell.d
open (T,"<cell.d");
while ($line=<T>) {
  chomp($line);
  @splitline=split(/\s+/, $line);
  @result[1]= "$splitline[2]";
  @result[2]= "$splitline[3]";
  @result[3]= "$splitline[4]";
  @result[4]= "$splitline[5]";
  @result[5]= "$splitline[6]";
  @result[6]= "$splitline[7]";
  @result[7]= "$splitline[8]";
  @result[8]= "$splitline[9]";
  @result[9]= "$splitline[10]";
}
close (T);
  $l1=sqrt($result[1]**2+$result[2]**2+$result[3]**2);
  $l2=sqrt($result[4]**2+$result[5]**2+$result[6]**2);
  $l3=sqrt($result[7]**2+$result[8]**2+$result[9]**2);

print S "$lz $ene $sxx $syy $szz $l1 $l2 $l3\n";

$lz=$lz+$lzinc;
}
close(S);

```

### 10.3 シーケンシャルピクチャの作製

適当なステップ数 (例えば 100 ステップ) 走らせるような SETDAT を書いておき, 初期配置を CONFIG に用意する.

シェルスクリプト

```
#!/usr/bin/perl
system ("rm -rf conf*cfg");
for ($count=1; $count<=100; $count++) {
    $num=sprintf("%03d",$count);
    if ($count==1) {
        $contvalue="no";
    } else {
        $contvalue="yes";
    }

    # Read SETDAT and change "continue_md" key
    open(C, "<SETDAT");
    open(F, ">SETDAT.TMP");
    while ($line=<C>) {
        chomp($line);
        @splitline=split(/\s+/, $line);
        if ($splitline[0] eq 'continue_md') {
            print F "continue_md $contvalue\n";
        } else {
            print F "$line\n";
        }
    }
    close(C);
    close(F);
    system ("rm SETDAT"); system ("mv SETDAT.TMP SETDAT");
    system ("./ymd");
    system ("cp conf.cfg conf$num.cfg");
    system ("rm CONT");
    system ("cp LAST CONT");
}
```

を走らせると, 100 ステップ×100 回の MD 計算を行い, 100 ステップ終わる毎に conf\*.jpg に原子配置が書き出される. atomeye conf001.jpg とすればスナップショットを見ることができる (Del キーで forward, Ins キーで backward).