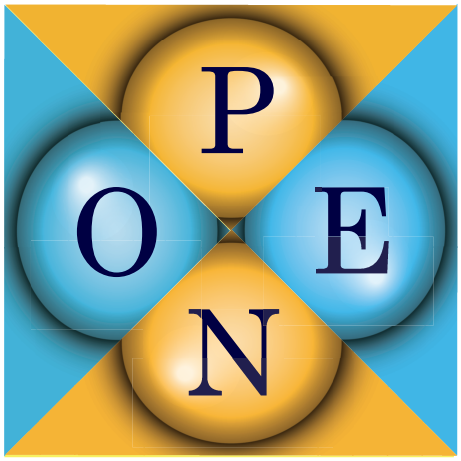
OpenMX Ver. 3.7 ユーザーマニュアル



開発貢献者

尾崎泰助 (北陸先端科学技術大学院大学)

木野日織 (物質・材料研究機構)

J. Yu (SNU)

M. J. Han (KAIST)

大淵真理 (富士通研究所)

石井史之 (金沢大学)

澤田啓介 (東京大学)

久保田雄士 (金沢大学)

大脇創 (日産総合研究所)

H. Weng (CAS)

豊田雅之 (大阪大学)

奥野幸洋 (富士フイルム)

R. Perez (UAM)

P.P. Bell (UAM)

T.V.T Duy (東京大学)

Yang Xiao (NUAA)

伊藤篤史 (核融合科学研究所)

寺倉清之 (産業技術総合研究所)

平成 26 年 1 月 10 日

目 次

1 OpenMXについて

OpenMX（Open source package for Material eXplorer）は密度汎関数理論（DFT）[1]、ノルム保存型擬ポテンシャル [19, 20, 21, 22, 23] および擬原子基底関数 [28] に基づき、原子レベルから物質の第一原理シミュレーションを実行するためのソフトウェア・パッケージです。OpenMX で用いられている計算手法、アルゴリズム、またそのプログラム上での実装は並列コンピュータ上での MPI 並列や OpenMP/MPI ハイブリッド並列による大規模第一原理電子状態計算を実現するために、入念に設計されています。OpenMX において実現された DFT の効率的な実装により、炭素系材料、生体分子、磁性体およびナノスケール電気伝導体など幅広い物質の幾何構造、電子構造、また磁気構造などを第一原理的に計算することが可能です。数百コアを有する並列コンピュータを使用することにより、千個の原子で構成される系を通常の対角化手法を用いて扱うことができます。さらに数千コアを有する並列コンピュータを使用すれば、一万個以上の原子から構成される系の第一原理による電子状態計算さえも、OpenMX に実装されている O(N) 法を用いて実行可能です。多くの元素に対して最適化された擬ポテンシャルおよび基底関数がデータベースとして整備されており、またその精度はベンチマーク計算によって検証されています。そのためユーザは自分でこれらのデータを準備する必要がなく、速やかに各自のシミュレーションを開始することができます。OpenMX には磁気特性、誘電特性、電気伝導特性などの物質特性を計算するための充実した機能が実装されており、ナノスケール物質科学における有用かつ複雑な物質を、量子力学に基づいてより深く理解するための強力な理論ツールとして、幅広い活用が期待されます。これまでの応用計算の事例は OpenMX の Web サイト (http://www.openmx.-square.org/) に記載されていますので、参考にして下さい。OpenMX の開発は、2000 年に尾崎グループにより開始され、その後、本マニュアルの冒頭のリストに挙げられている数多くの開発者が本オープンソースパッケージの開発に貢献してきました。プログラム・パッケージとソースコードは GNU-GPL(一般公的使用許諾) [59] に準じて配布されており、OpenMX の Web サイトからダウンロードすることができます。

OpenMX Ver. 3.7 の特徴と機能は以下のようになります。

- クラスタ計算法、バンド計算法、O(N) 法、低次スケーリング法による全エネルギーと力の計算
- 交換相関エネルギーに対する局所密度近似（LDA, LSDA）[2, 3, 4] および一般化勾配近似（GGA）[5]
- LDA(GGA)+U 法 [16]
- ノルム保存型擬ポテンシャル [2, 20, 21, 23]
- 変分的に最適化された擬原子基底関数 [28]
- 擬ポテンシャル法の枠組みでの完全および半相対論的取り扱い [10, 19, 13]
- ノンコリニア DFT 法 [6, 7, 8, 9]
- ノンコリニア DFT 法におけるスピン磁気モーメントおよび軌道磁気モーメントの方位制御 [11]
- Berry 位相法による巨視的分極率の計算 [12]

- 分割統治法（DC 法）[37] と Krylov 部分空間法による O(N) 固有値ソルバ
- ELPA [26] による並列固有値ソルバ
- 単純混合法、RMM-DIIS 法 [40]、GR-Pulay 法 [39], 、Kerker 法 [41]、Kerker 因子による重み付き RMM-DIIS 法 [40] による電荷密度の混合
- 交換結合パラメータ [14, 15] の計算
- 有効遮蔽媒質法 (ESM) [81, 84]
- 走査型トンネル顕微鏡（STM）シミュレーション [52]
- Nudged Elastic Band 法（NEB 法）[53]
- 電荷ドーピング
- のこぎり波による均一電界の導入
- 完全および制限付き構造最適化
- 非平衡グリーン関数（NEGF）法による電気伝導計算 [54]
- 最大局所化ワニエ関数の計算
- NVE アンサンブル分子動力学
- 速度スケーリング法 [17] および Nose-Hoover 法 [18] による NVT アンサンブル分子動力学
- マリケン (Mulliken) 法、ボロノイ (Voronoi) 法、ESP 法による電荷・スピン数の解析
- 波動関数および電子・スピン密度分布の解析
- バンド計算によるバンド分散の解析
- 状態密度（DOS）および射影 DOS の計算
- 柔軟なデータ書式 (flexible data format) による入力ファイル
- 電荷密度などを可視化するための XCrySDen へのインタフェース [61]
- 完全な動的メモリ割当て
- メッセージ・パッシング・インタフェース (MPI) による並列計算
- OpenMP による並列計算
- 開発者向けの利便性の高いユーザーインタフェース

コリニアおよびノンコリニア DFT 法のそれぞれはスカラーおよび完全相対論的擬ポテンシャルに対応して実装されています。またスピンおよび軌道磁気モーメントを制御するために制限ノンコリニア DFT 法が実装されています。これらの手法は複雑なノンコリニア磁気構造やスピン軌道相互作用を調べるのに役に立ちます。通常の対角化による計算は、数千コアまでの並列化が実現できる ELPA に基く並列固有値ソルバ [26] によって行われます。この機能により、通常の対角化法を用いて千個の原子からなる系の計算が可能になります。この高並列化対角化手法によってクラスタ、分子、スラブおよび固体に対する大規模計算が可能ですが、線形スケーリング法および低次スケーリング法も固有値ソルバとして利用可能です。これらの低次スケーリング法に対しては計算精度と計算効率に対して注意深い検討を行うことで、1 万原子を越える系をも取り扱うことが可能です。また OpenMX Ver. 3.7 の重要な機能の一つとして、NEGF 法に基づく電気伝導計算が、コリニア DFT 法だけでなく相対論的擬ポテンシャルおよび磁気モーメント制約法を用いたノンコリニア DFT 法にも対応していることが挙げられます。

OpenMX の開発は継続して行われています。本オープンソースコードの発展に寄与して頂ける開発者の方のご参加を歓迎致します。

2 インストール

2.1 必須ライブラリ

OpenMX は、以下の 3 つのライブラリがインストール済みの Linux 環境にインストールすることができます。

- LAPACK (および BLAS) (http://www.netlib.org/)
- FFTW (http://www.fftw.org/)
- MPICH2 や OpenMPI などの MPI ライブラリ

これらのライブラリ・パッケージが対象コンピュータにインストールされていない場合、OpenMX をインストールする前にこれらをインストールする必要があります。MPICH2 や OpenMPI などの MPI ライブラリは、OpenMX Ver. 3.7 のインストールに必須です。MPI ライブラリが無い場合、OpenMX Ver. 3.7 をインストールする次のステップに進むことはできません。上記のライブラリ・パッケージがインストール済みの場合は、次のインストール手順に進んでください。openmx3.7.tar.gz をダウンロード後、以下のコマンドで解凍します。

```
% tar zxvf openmx3.7.tar.gz
```

解凍すると、「opnemx3.7」ディレクトリの下に「source」、「work」、「DFT_DATA13」の 3 つのディレクトリが作成されます。これらのディレクトリにはそれぞれソースコード、入力ファイル、最適化された Ver. 2013 の擬原子基底関数および擬ポテンシャルのデータファイルが含まれます。

2.2 シリアル版

OpenMX Ver. 3.7 ではシリアル版のインストールはサポートされていません。

2.3 MPI 版

MPI 版の OpenMX をインストールするには、「source」ディレクトリに移動し、「makefile」を編集し、**CC**、**FC**、**LIB** を設定することでコンパイラおよびライブラリを指定します。「makefile」のデフォルトの **CC**、**FC**、**LIB** 設定は以下の通りです。

```
CC    = mpicc -Dnoomp -O3 -I/usr/local/include
FC    = mpif90 -Dnoomp -O3 -I/usr/local/include
LIB   = -L/usr/local/lib -lfftw3 -llapack -lblas -lg2c -static
```

CC と **FC** は、それぞれ C と FORTRAN のコンパイラを指定し、**LIB** はリンクするライブラリを指定します。Ver. 3.6 までは **FC** の指定は不要でしたが、ELPA に基づく並列固有値ソルバ [26] の導入に伴い、Ver. 3.7 からはこれが必要となります。OpenMP が無い環境では「-Dnoomp」オプションを指定してください。コンパイルとリンクが正しく行われ、 十分に最適化された実行ファイルを作成するには、

実行環境に適した **CC**、**FC**、**LIB** の設定を行うことが必要です。これらのオプションの設定後、以下のコマンドよりインストールを実行します。

```
% make install
```

コンパイルが正常に完了すると、「openmx」という実行ファイルが「work」ディレクトリの下に作成されます。OpenMX の実行効率を上げるために、必要に応じて実行環境に合わせてコンパイラやコンパイラオプションを変更することで、最適化された実行ファイルを生成することができます。「source」ディレクトリの「makefile」ファイル内には、**CC**、**FC**、**LIB** の設定例がいくつか用意されていますので参考にしてください。

2.4 OpenMP/MPI ハイブリッド並列版

OpenMP/MPI ハイブリッドバージョンを生成する場合には、「source」ディレクトリの「makefile」ファイル内の **CC** および **FC** に、OpenMP 並列化のコンパイラオプションを追加するだけです。OpenMP/MPI バージョンをインストールするには、「source」ディレクトリに移動し、「makefile」内の **CC**、**FC**、**LIB** を以下の例のように指定します。

icc の場合

```
CC    = mpicc -openmp -O3 -I/usr/local/include
FC    = mpif90 -openmp -O3 -I/usr/local/include
LIB   = -L/usr/local/lib -lfftw3 -llapack -lblas -lg2c -static
```

pgcc の場合

```
CC    = mpicc -mp -O3 -I/usr/local/include
FC    = mpif90 -mp -O3 -I/usr/local/include
LIB   = -L/usr/local/lib -lfftw3 -llapack -lblas -lg2c -static
```

OpenMP のコンパイラオプションはコンパイラに依存します。また、古いバージョンの icc および pgcc コンパイラは OpenMP のコンパイラオプションをサポートしないことに注意してください。**CC**、**FC**、**LIB** を適切に指定した後、下記のコマンドでインストールを実行します。

```
% make install
```

コンパイルが正常に終了すると、「work」ディレクトリに「openmx」実行ファイルが生成されます。OpenMX の実行効率を上げるために、必要に応じて実行環境に合わせてコンパイラやコンパイラオプションを変更することで、最適化された実行ファイルを生成することができます。

2.5 FFTW3

Ver. 3.6 までの旧バージョンは FFTW2 と FFTW3 の両方に対応していましたが、OpenMX Ver. 3.7 では FFTW3 のみが使用可能です。makefile 中で以下のように指定することで FFTW3 とリンクすることができます。

```
LIB      = -L/usr/local/lib -fftw3 -llapack -lblas -lg2c -static
```

2.6 その他のオプション

2.6.1 -Dblaswrap および-lI77

実行環境によっては、-Dblaswrap および-lI77 というオプションを追加しなくてはならない場合があります。ただし、何故これらのオプションに依存する場合があるのかは完全には分かっていません。これらのオプションを必要とする場合は、**CC**、**FC**、**LIB** を以下のように設定します。

```
CC    = mpicc -openmp -O3 -Dblaswrap -I/usr/local/include
FC    = mpif90 -mp -openmp -Dblaswrap -I/usr/local/include
LIB   = -L/usr/local/lib -lfftw3 -llapack -lblas -lg2c -lI77 -static
```

2.6.2 -Df77, -Df77_, -Df77__, -DF77, -DF77_, -DF77__

lapack および blas ライブラリをリンクする際に、ルーチンの指定がマシン環境に依存する場合があります。大文字、小文字、アンダースコアがある場合とそうでない場合があります。ご利用の環境で正しく lapack および blas ルーチンを指定するには、-Df77、-Df77_、-Df77__、-DF77、-DF77_、-DF77__のいずれかのオプションを指定します。lapack ルーチンを呼び出す際に大文字が必要な場合には「F」が入ったオプションを使用し、さらにアンダースコアを、なし、「_」、あるいは「__」から選びます。デフォルトでは-Df77_（小文字＋アンダースコア一つ）です。

2.6.3 -Dnosse

O(N) クリロフ (Krylov) 部分空間法のルーチン（Krylov.c）は、ストリーミング SIMD 拡張命令（SSE）を使用して最適化されているため、デフォルトでは SSE を利用するようにソースコードがコンパイルされます。お使いのプロセッサが SSE に対応していない場合、-Dnosse オプションを **CC** に含めてください。

2.6.4 -Dkcomp

富士通 (株) の SPARC プロセッサをご利用の場合は、**CC** および **FC** コンパイラオプションに-Dkcomp を含めてください。

2.7 プラットフォーム

OpenMX Ver. 3.7 は、これまで以下のプラットフォームで正常に作動することが確認されています。

- Sandy Bridge Xeon クラスタ
- Opteron クラスタ
- CRAY-XC30
- 富士通 FX10
- 理化学研究所のスーパーコンピュータ「京」

2.8 インストールの際のヒント

OpenMX のインストールにおける問題のほとんどは、LAPACK、BLAS およびこれらの代替ライブラリへのリンクが原因です。多くの場合において ACML あるいは MKL をリンクすることをお勧めしますが、計算速度および安定性において ACML が MKL よりやや優位性があります。これらのライブラリとリンクする例は「source」ディレクトリの「makefile」内に記されています。

いくつかのポピュラーなプラットフォーム上でのインストールに関するヒントを以下に記します。OpenMX は、C と FORTRAN のコンパイラ、および LAPACK、BLAS、FFT ライブラリを必要とします。更に、C コンパイラはリンクのために使用されるため、コンパイラに対応する FORTRAN ライブラリを明示的に指定しなければなりません。下記に一般的なコンパイラおよび LAPACK、BLAS ライブラリにおけるインストール設定例を記します。ここで、FFT ライブラリは/usr/local/fftw3/にインストールされていると仮定します。

- Intel の C および FORTRAN コンパイラ（icc、ifort)、LAPACK と BLAS には MKL ライブラリを使用

  MKLROOT=/opt/intel/mkl
  CC=mpicc -O3 -xHOST -openmp -I/usr/local/fftw3/include -I/$MKLROOT/include
  FC=mpiifort -O3 -xHOST -openmp -I/$MKLROOT/include
  LIB= -L/usr/local/fftw3/lib -lfftw3 -L/$MKLROOT/lib/intel64/ -lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core -lpthread -lifcore

- PGI の C および FORTRAN コンパイラ（pgccpgCC、pgf77、pgf90)、LAPACK と BLAS には ACML ライブラリを使用

  CC=mpicc -fast -mp -Dnosse -I/usr/local/fftw3/include -I/usr/local/acml/gnu64/include
  FC=mpif90 -fast -mp -I/usr/local/acml/gnu64/include
  LIB= -L/usr/local/fftw3/lib -lfftw3 /usr/local/acml/gnu64/lib/libacml.a /usr/lib64/libg2c.a -pgf90libs

  重要：PGI C コンパイラで-mp オプションを使って OpenMP を有効にする場合は、プリプロセッサオプション-Dnosse を指定しなければなりません。

- GNU C および FORTRAN コンパイラ（gcc、g++、gfortran）、 LAPACK と BLAS には MKL ライブラリを使用

  MKLROOT=/opt/intel/mkl
  CC=mpicc -O3 -ffast-math -fopenmp -I/usr/local/fftw3/include -I/$MKLROOT/include
  FC=mpif90 -O3 -ffast-math -fopenmp -I/$MKLROOT/include
  LIB= -L/usr/local/fftw3/lib -lfftw3 -L/$MKLROOT/lib/intel64/ -lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core -lpthread -lgfortran

- GNU C および FORTRAN コンパイラ（gcc、g++、gfortran）、 LAPACK と BLAS には ACML ライブラリを使用

  CC=mpicc -O3 -ffast-math -fopenmp -I/usr/local/fftw3/include -I/usr/local/acml/gnu64/include
  FC=mpif90 -O3 -ffast-math -fopenmp -I/usr/local/acml/gnu64/include
  LIB= -L/usr/local/fftw3/lib -lfftw3 /usr/local/acml/gnu64/lib/libacml.a -lgfortran

その他のコンパイラと LAPACK、BLAS ライブラリの組み合わせも同様に可能です。下記のコマンドを使用することで、MPI が利用するコンパイラ（Intel、PGI、GNU 等の）についての情報を表示することができます。

```
%mpicc -compile-info (MPICH の場合)
%mpicc -help  (OpenMPI の場合)
```

C コンパイラが FORTRAN ライブラリの保存場所を見つけることができない場合、次のようなリンクエラーが生じることがあります。

Intel コンパイラの場合：

```
/usr/bin/ld: cannot find -lifcore
```

PGI コンパイラの場合：

```
/usr/bin/ld: cannot find -lpgf90
```

あるいは

```
-lpgf90_rpm1, -lpgf902, -lpgf90rtl, -lpgftnrtl
```

（-pgf90libs は、これらへのショートカットにすぎないため）

GNU コンパイラの場合：

```
/usr/bin/ld: cannot find -lgfortran
```

このリンクに関する問題を解決するには、以下の手順で FORTRAN ライブラリを明示的に指定する必要があります。まず FORTRAN コンパイラの保存場所は次のコマンドで確認することができます。

```
%which ifort（Intel コンパイラの場合）
/opt/intel/fce/10.0.026/bin/ifort

%which pgf90（PGI コンパイラの場合）
/opt/pgi/linux86-64/7.0/bin/pgf90

%which gfortran（GNU コンパイラの場合）
/usr/bin/gfortran
```

次に、FORTRAN ライブラリは通常、上記/bin ディレクトリの親ディレクトリの下にある/lib ディレクトリにあるため、次のように LIB を指定します。

```
LIB= ... -L/opt/intel/fce/10.0.026/lib -lifcore（Intel コンパイラの場合）
LIB= ... -L/opt/pgi/linux86-64/7.0/lib -pgf90libs（PGI コンパイラの場合）
LIB= ... -L/usr/lib -lgfortran（GNU コンパイラの場合）
```

3　テスト計算

インストールが正常に終了したら、「work」ディレクトリに移動し、「Methane.dat」ファイルを入力ファイルとして下記のようにして「openmx」を実行して下さい。

```
% mpirun -np 1 openmx Methane.dat > met.std &
```

OpenMP/MPI ハイブリッドバージョンをお使いの場合は下記のコマンドを使います。

```
% mpirun -np 1 openmx Methane.dat -nt 1 > met.std &
```

計算で用いた「Methane.dat」は、メタン分子の SCF 計算を実行するための入力ファイルです。構造最適化は行いません。この計算は 2.6 GHz Xeon 搭載マシンを使った場合、12 秒程度で終了する筈ですが、実際の実行時間はコンピュータ環境に依存します。計算が正常に終了すると、次の 11 個のファイルと 1 個のディレクトリが「work」ディレクトリに作成されます。

met.std	SCF 計算の標準出力
met.out	入力ファイルと標準出力
met.xyz	最終的な幾何構造
met.ene	各 MD ステップにおける計算値
met.md	各 MD ステップにおける幾何構造
met.md2	最終 MD ステップにおける幾何構造
met.cif	Material Stuido 用の初期構造の cif ファイル
met.tden.cube	Gaussian cube 形式の全電子密度
met.v0.cube	Gaussian cube 形式の Kohn-Sham ポテンシャル
met.vhart.cube	Gaussian cube 形式の Hartree ポテンシャル
met.dden.cube	原子密度から計算した差電子密度
met_rst/	再スタートファイルを保存するディレクトリ

標準出力へ出力されるデータは「met.std」ファイルに格納されます。これは SCF 計算時の計算過程を知るために役立ちます。「met.out」ファイルには、全エネルギー、力、Kohn-Sham 固有値、Mulliken 電荷、SCF 計算の収束履歴、および計算時間などの結果が記載されています。参考のために「met.out」ファイルの一部を下記に抜粋します。11 回の SCF 反復でエネルギー固有値が 1.0e-10 ハートリー (Hartree) 内に収束していることが分かります。

```
*******************************************************
*******************************************************
                  SCF history at MD= 1
*******************************************************
*******************************************************

   SCF=   1  NormRD=  1.000000000000  Uele= -3.523143659974
```

```
 SCF=   2  NormRD=  0.567253699744  Uele= -4.405605131921
 SCF=   3  NormRD=  0.103433490729  Uele= -3.982266241934
 SCF=   4  NormRD=  0.024234990593  Uele= -3.906896836134
 SCF=   5  NormRD=  0.011006215697  Uele= -3.893084558820
 SCF=   6  NormRD=  0.006494145332  Uele= -3.890357113476
 SCF=   7  NormRD=  0.002722267527  Uele= -3.891669816209
 SCF=   8  NormRD=  0.000000672350  Uele= -3.889285164733
 SCF=   9  NormRD=  0.000000402419  Uele= -3.889285102456
 SCF=  10  NormRD=  0.000000346348  Uele= -3.889285101128
 SCF=  11  NormRD=  0.000000515395  Uele= -3.889285101063
```

また全エネルギー、化学ポテンシャル、Kohn-Sham 固有値、Mulliken 電荷、双極子モーメント、力、規格化座標、計算時間などが「met.out」ファイルに以下のように出力されます。

```
*********************************************************
        Total energy (Hartree) at MD = 1
*********************************************************

  Uele.         -3.889285101063

  Ukin.          5.533754016241
  UH0.         -14.855520072374
  UH1.           0.041395625260
  Una.          -5.040583803800
  Unl.          -0.134640939010
  Uxc0.         -1.564720823137
  Uxc1.         -1.564720823137
  Ucore.         9.551521413583
  Uhub.          0.000000000000
  Ucs.           0.000000000000
  Uzs.           0.000000000000
  Uzo.           0.000000000000
  Uef.           0.000000000000
  UvdW           0.000000000000
  Utot.         -8.033515406373

  Note:

  Utot = Ukin+UH0+UH1+Una+Unl+Uxc0+Uxc1+Ucore+Uhub+Ucs+Uzs+Uzo+Uef+UvdW
```

```
  Uene:   band energy
  Ukin:   kinetic energy
  UH0:    electric part of screened Coulomb energy
  UH1:    difference electron-electron Coulomb energy
  Una:    neutral atom potential energy
  Unl:    non-local potential energy
  Uxc0:   exchange-correlation energy for alpha spin
  Uxc1:   exchange-correlation energy for beta spin
  Ucore:  core-core Coulomb energy
  Uhub:   LDA+U energy
  Ucs:    constraint energy for spin orientation
  Uzs:    Zeeman term for spin magnetic moment
  Uzo:    Zeeman term for orbital magnetic moment
  Uef:    electric energy by electric field
  UvdW:   semi-empirical vdW energy

  (see also PRB 72, 045121(2005) for the energy contributions)



  Chemical potential (Hartree)       0.000000000000

***********************************************************
***********************************************************
           Eigenvalues (Hartree) for SCF KS-eq.
***********************************************************
***********************************************************

   Chemical Potential (Hartree) =   0.00000000000000
   Number of States             =   8.00000000000000
   HOMO =  4
   Eigenvalues
                Up-spin            Down-spin
         1  -0.69897190537228  -0.69897190537228
         2  -0.41522646150979  -0.41522646150979
         3  -0.41522645534084  -0.41522645534084
         4  -0.41521772830844  -0.41521772830844
         5   0.21218282298348   0.21218282298348
```

```
          6   0.21218282358344   0.21218282358344
          7   0.21227055734372   0.21227055734372
          8   0.24742493684297   0.24742493684297

************************************************************
************************************************************
                  Mulliken populations
************************************************************
************************************************************

  Total spin S =  0.000000000000

                     Up spin      Down spin     Sum           Diff
     1    C       2.509755704  2.509755704   5.019511408   0.000000000
     2    H       0.372561098  0.372561098   0.745122197   0.000000000
     3    H       0.372561019  0.372561019   0.745122038   0.000000000
     4    H       0.372561127  0.372561127   0.745122254   0.000000000
     5    H       0.372561051  0.372561051   0.745122102   0.000000000

 Sum of MulP: up   =     4.00000 down          =     4.00000
              total=     8.00000 ideal(neutral)=     8.00000

  Decomposed Mulliken populations

    1    C          Up spin      Down spin     Sum           Diff
           multiple
  s           0    0.681752967  0.681752967   1.363505935   0.000000000
   sum over m      0.681752967  0.681752967   1.363505935   0.000000000
   sum over m+mul  0.681752967  0.681752967   1.363505935   0.000000000
  px          0    0.609349992  0.609349992   1.218699985   0.000000000
  py          0    0.609302752  0.609302752   1.218605504   0.000000000
  pz          0    0.609349993  0.609349993   1.218699985   0.000000000
   sum over m      1.828002737  1.828002737   3.656005474   0.000000000
   sum over m+mul  1.828002737  1.828002737   3.656005474   0.000000000

    2    H          Up spin      Down spin     Sum           Diff
           multiple
  s           0    0.372561098  0.372561098   0.745122197   0.000000000
   sum over m      0.372561098  0.372561098   0.745122197   0.000000000
```

```
   sum over m+mul  0.372561098  0.372561098   0.745122197   0.000000000

    3    H          Up spin      Down spin     Sum           Diff
           multiple
  s           0    0.372561019  0.372561019   0.745122038   0.000000000
   sum over m      0.372561019  0.372561019   0.745122038   0.000000000
   sum over m+mul  0.372561019  0.372561019   0.745122038   0.000000000

    4    H          Up spin      Down spin     Sum           Diff
           multiple
  s           0    0.372561127  0.372561127   0.745122254   0.000000000
   sum over m      0.372561127  0.372561127   0.745122254   0.000000000
   sum over m+mul  0.372561127  0.372561127   0.745122254   0.000000000

    5    H          Up spin      Down spin     Sum           Diff
           multiple
  s           0    0.372561051  0.372561051   0.745122102   0.000000000
   sum over m      0.372561051  0.372561051   0.745122102   0.000000000
   sum over m+mul  0.372561051  0.372561051   0.745122102   0.000000000

************************************************************
************************************************************
                   Dipole moment (Debye)
************************************************************
************************************************************

 Absolute D        0.00000071

                      Dx              Dy              Dz
 Total             0.00000046      0.00000000     -0.00000054
 Core              0.00000000      0.00000000      0.00000000
 Electron          0.00000046      0.00000000     -0.00000054
 Back ground      -0.00000000     -0.00000000     -0.00000000

************************************************************
************************************************************
       xyz-coordinates (Ang) and forces (Hartree/Bohr)
************************************************************
************************************************************
```

```
<coordinates.forces
  5
    1     C     0.00000   0.00000   0.00000   0.000000000327 -0.000...
    2     H    -0.88998  -0.62931   0.00000  -0.064883705001 -0.045...
    3     H     0.00000   0.62931  -0.88998   0.000000043463  0.045...
    4     H     0.00000   0.62931   0.88998   0.000000045939  0.045...
    5     H     0.88998  -0.62931   0.00000   0.064883635459 -0.045...
coordinates.forces>

************************************************************
************************************************************
       Fractional coordinates of the final structure
************************************************************
************************************************************

     1      C     0.00000000000000   0.00000000000000   0.00000000000000
     2      H     0.91100190000000   0.93706880000000   0.00000000000000
     3      H     0.00000000000000   0.06293120000000   0.91100190000000
     4      H     0.00000000000000   0.06293120000000   0.08899810000000
     5      H     0.08899810000000   0.93706880000000   0.00000000000000

************************************************************
************************************************************
               Computational Time (second)
************************************************************
************************************************************

   Elapsed.Time.         11.725

                                Min_ID    Min_Time        Max_ID    Max_Time
   Total Computational Time =      0        11.725           0        11.725
   readfile                 =      0         8.987           0         8.987
   truncation               =      0         0.155           0         0.155
   MD_pac                   =      0         0.000           0         0.000
   OutData                  =      0         0.452           0         0.452
   DFT                      =      0         2.130           0         2.130

*** In DFT ***
```

```
Set_OLP_Kin             =     0        0.127          0        0.127
Set_Nonlocal            =     0        0.104          0        0.104
Set_ProExpn_VNA         =     0        0.132          0        0.132
Set_Hamiltonian         =     0        0.741          0        0.741
Poisson                 =     0        0.351          0        0.351
Diagonalization         =     0        0.004          0        0.004
Mixing_DM               =     0        0.000          0        0.000
Force                   =     0        0.200          0        0.200
Total_Energy            =     0        0.296          0        0.296
Set_Aden_Grid           =     0        0.022          0        0.022
Set_Orbitals_Grid       =     0        0.026          0        0.026
Set_Density_Grid        =     0        0.120          0        0.120
RestartFileDFT          =     0        0.003          0        0.003
Mulliken_Charge         =     0        0.000          0        0.000
FFT(2D)_Density         =     0        0.000          0        0.000
Others                  =     0        0.003          0        0.003
```

出力ファイル「met.tden.cube」、「met.v0.cube」、「met.vhart.cube」、および「met.dden.cube」には、それぞれ全電子密度、Kohn-Sham ポテンシャル、Hartree ポテンシャル、および構成孤立原子の電子密度の重ね合わせを基準とした差電子密度が Gaussian cube 形式で書き出されています。Gaussian cube 形式は広く使われているグリッド形式であるため、Molekel [60] や XCrySDen [61] などの無料の分子モデリングソフトウェアを用いて可視化することができます。後述の節にて可視化の例を示します。

4　自動実行のテスト

「テスト計算」の章で説明した計算に加え、OpenMX の主な機能が正常にインストールされているか確認したい場合には、自動実行のテストを実施することをお薦めします。この自動実行テストを行うには、以下のコマンドで OpenMX を実行します。

MPI 並列化の場合：

```
% mpirun -np 8 openmx -runtest
```

OpenMP/MPI 並列化の場合：

```
% mpirun -np 8 openmx -runtest -nt 2
```

並列計算の実行の際に mpirun の他のオプションを指定することもできます。このテストでは、OpenMX は 14 個の入力テストファイルを計算し、結果を「work/input_example」に格納されている参照データと比較します。比較結果（全エネルギーおよび力の差の絶対値）は「work」ディレクトリの「runtest.result」というファイルに格納されます。参照データは 2.6 GHz Xeon シングルプロセッサのコンピュータで計算されたものです。絶対差分が小数点以下 7 桁以内であればインストールは正常に行われたと判断できます。自動実行テストの結果、生成された「runtest.result」の例を以下に示します。

1	input_example/Benzene.dat	Elapsed time(s)= 4.78	diff Utot= 0.000000000000	diff Force= 0.000000000002
2	input_example/C60.dat	Elapsed time(s)= 14.96	diff Utot= 0.000000000019	diff Force= 0.000000000004
3	input_example/CO.dat	Elapsed time(s)= 9.86	diff Utot= 0.000000000416	diff Force= 0.000000000490
4	input_example/Cr2.dat	Elapsed time(s)= 10.70	diff Utot= 0.000000000000	diff Force= 0.000000000044
5	input_example/Crys-MnO.dat	Elapsed time(s)= 19.98	diff Utot= 0.000000004126	diff Force= 0.000000001888
6	input_example/GaAs.dat	Elapsed time(s)= 26.39	diff Utot= 0.000000001030	diff Force= 0.000000000007
7	input_example/Glycine.dat	Elapsed time(s)= 5.48	diff Utot= 0.000000000001	diff Force= 0.000000000000
8	input_example/Graphite4.dat	Elapsed time(s)= 5.00	diff Utot= 0.000000002617	diff Force= 0.000000015163
9	input_example/H2O-EF.dat	Elapsed time(s)= 4.88	diff Utot= 0.000000000000	diff Force= 0.000000000113
10	input_example/H2O.dat	Elapsed time(s)= 4.60	diff Utot= 0.000000000008	diff Force= 0.000000013375
11	input_example/HMn.dat	Elapsed time(s)= 13.44	diff Utot= 0.000000000001	diff Force= 0.000000000001
12	input_example/Methane.dat	Elapsed time(s)= 3.64	diff Utot= 0.000000000001	diff Force= 0.000000002263
13	input_example/Mol_MnO.dat	Elapsed time(s)= 9.43	diff Utot= 0.000000003714	diff Force= 0.000000000540
14	input_example/Ndia2.dat	Elapsed time(s)= 5.67	diff Utot= 0.000000000004	diff Force= 0.000000000001

Total elapsed time (s) 138.79

この比較は同一 Xeon クラスタ上で、8 MPI プロセスと 2 OpenMP スレッドを用いて行われました。浮動小数点の演算はコンピュータ環境だけでなく並列計算で使用されたプロセッサ数等にも依存するため、上記の例では、同じコンピュータを使っているのにも関わらず差が現れているのが分かります。各ジョブの経過時間も出力されているため、環境によって計算スピードの差を比較することができます。「work/input_example」ディレクトリには、複数のプラットフォームで生成された「runtest.result」ファイルがあります。

参照データファイルを御自身で作成する場合には、以下のコマンドによって OpenMX を実行してください。

```
% ./openmx -maketest
```

これにより OpenMX は「work/input_example」ディレクトリにある「*.dat」入力ファイルに対して、対応する「*.out」ファイルを同ディレクトリに生成します。新しい入力 dat ファイルを追加することで次回の自動実行テスト時に、対応する「*.out」を利用することができますが、この手続きにより以前の「work/input_example」にある出力ファイルは上書きされてしまうことに注意してください。開発者やヘビーユーザがコードの信頼性を確認したい場合には「自動フォース・テスター」および「自動メモリリーク・テスター」の章もご参照ください。

5　大規模な系の自動実行テスト

より大規模な系のベンチマーク計算によって計算機性能を知りたい場合、以下のコマンドにて、テスト計算を自動実行することができます。

MPI 並列化の場合：

```
% mpirun -np 128 openmx -runtestL
```

OpenMP/MPI 並列化の場合：

```
% mpirun -np 128 openmx -runtestL -nt 2
```

これらのコマンドを実行すると、OpenMX は 16 個のテスト入力ファイルを実行し、その結果を「work/large_example」にある参照データと比較します。比較結果（全エネルギーおよび力の絶対差分）は、「work」ディレクトリの「runtestL.result」ファイルに格納されます。参照データは、2.6 GHz Xeon クラスタマシン上で 16 個の MPI プロセスを使用して計算されたものです。絶対差分が小数点以下 7 桁以内であればインストレーションは正常に行われたと判断できます。例として、自動実行テストで生成された「runtestL.result」を以下に示します。

1	large_example/5_5_13COb2.dat	Elapsed time(s)= 39.43	diff Utot= 0.000000000013	diff Force= 0.000000000046
2	large_example/B2C62_Band.dat	Elapsed time(s)= 572.22	diff Utot= 0.000000000025	diff Force= 0.000000013928
3	large_example/CG15c-Kry.dat	Elapsed time(s)= 40.71	diff Utot= 0.000000002112	diff Force= 0.000000001090
4	large_example/DIA512-1.dat	Elapsed time(s)= 37.93	diff Utot= 0.000000169524	diff Force= 0.000000033761
5	large_example/FeBCC.dat	Elapsed time(s)= 81.55	diff Utot= 0.000000000649	diff Force= 0.000000001349
6	large_example/GEL.dat	Elapsed time(s)= 47.05	diff Utot= 0.000000000066	diff Force= 0.000000000002
7	large_example/GFRAG.dat	Elapsed time(s)= 24.05	diff Utot= 0.000000000122	diff Force= 0.000000000015
8	large_example/GGFF.dat	Elapsed time(s)= 639.31	diff Utot= 0.000000000051	diff Force= 0.000000000243
9	large_example/MCCN.dat	Elapsed time(s)= 53.72	diff Utot= 0.000000009994	diff Force= 0.000000016474
10	large_example/Mn12_148_F.dat	Elapsed time(s)= 76.58	diff Utot= 0.000000000096	diff Force= 0.000000000090
11	large_example/N1C999.dat	Elapsed time(s)= 97.56	diff Utot= 0.000000006902	diff Force= 0.000000007356
12	large_example/Ni63-O64.dat	Elapsed time(s)= 78.00	diff Utot= 0.000000000782	diff Force= 0.000000000047
13	large_example/Pt63.dat	Elapsed time(s)= 60.40	diff Utot= 0.000000002147	diff Force= 0.000000000059
14	large_example/SialicAcid.dat	Elapsed time(s)= 47.80	diff Utot= 0.000000000005	diff Force= 0.000000000003
15	large_example/ZrB2_2x2.dat	Elapsed time(s)= 143.16	diff Utot= 0.000000000030	diff Force= 0.000000000003
16	large_example/nsV4Bz5.dat	Elapsed time(s)= 104.20	diff Utot= 0.000000010770	diff Force= 0.000000000605

Total elapsed time (s) 2143.68

この結果は 128 個の MPI プロセスと 4 個の OpenMP スレッド（合計 256 コア）を用いて CRAY-XC30 上で計算したものです。この自動実行テストは大量のメモリを必要とするため、使用コア数が少ない場合にメモリの領域侵害を起こす場合があります。また上記の例では 256 コアを使用しても約 36 分の時間を要することが分かります。「大規模計算」の章にも別の大規模ベンチマーク計算の例の説明がありますので、参照して下さい。。

6　入力ファイル

6.1　例：メタン分子

「work」ディレクトリにある「Methane.dat」入力ファイルを例として下記に示します。入力ファイルは柔軟なデータ書式 (flexible data format) で記述されます。この形式ではキーワードの後に対応するパラメータを指定し、キーワードは順不同であり、空白行やコメントも自由に使用できます。下記の例では特定の様式で書かれていますが、キーワードやオプション（パラメータ）は、大文字、小文字の区別をしておらず、ユーザーの好みに応じて、大文字、小文字、あるいは大文字小文字混在で指定することができます。

```
#
# File Name
#

System.CurrrentDirectory         ./    # default=./
System.Name                      met
level.of.stdout                   1    # default=1 (1-3)
level.of.fileout                  1    # default=1 (0-2)

#
# Definition of Atomic Species
#

Species.Number       2
<Definition.of.Atomic.Species
 H   H5.0-s1          H_PBE13
 C   C5.0-s1p1        C_PBE13
Definition.of.Atomic.Species>

#
# Atoms
#

Atoms.Number         5
Atoms.SpeciesAndCoordinates.Unit   Ang # Ang|AU
<Atoms.SpeciesAndCoordinates
 1  C      0.000000    0.000000    0.000000     2.0  2.0
 2  H     -0.889981   -0.629312    0.000000     0.5  0.5
 3  H      0.000000    0.629312   -0.889981     0.5  0.5
 4  H      0.000000    0.629312    0.889981     0.5  0.5
 5  H      0.889981   -0.629312    0.000000     0.5  0.5
Atoms.SpeciesAndCoordinates>
Atoms.UnitVectors.Unit             Ang # Ang|AU
<Atoms.UnitVectors
  10.0   0.0   0.0
   0.0  10.0   0.0
   0.0   0.0  10.0
Atoms.UnitVectors>
```

```
#
# SCF or Electronic System
#

scf.XcType                  GGA-PBE     # LDA|LSDA-CA|LSDA-PW|GGA-PBE
scf.SpinPolarization         off        # On|Off|NC
scf.ElectronicTemperature  300.0        # default=300 (K)
scf.energycutoff           120.0        # default=150 (Ry)
scf.maxIter                 100         # default=40
scf.EigenvalueSolver        cluster     # DC|Cluster|Band
scf.Kgrid                  1 1 1        # means n1 x n2 x n3
scf.Mixing.Type           rmm-diis      # Simple|Rmm-Diis|Gr-Pulay|Kerker|Rmm-Diisk
scf.Init.Mixing.Weight     0.30         # default=0.30
scf.Min.Mixing.Weight      0.001        # default=0.001
scf.Max.Mixing.Weight      0.400        # default=0.40
scf.Mixing.History          7           # default=5
scf.Mixing.StartPulay       5           # default=6
scf.criterion             1.0e-10       # default=1.0e-6 (Hartree)

#
# MD or Geometry Optimization
#

MD.Type                      nomd       # Nomd|Opt|NVE|NVT_VS|NVT_NH
                                      # Constraint_Opt|DIIS
MD.maxIter                     1        # default=1
MD.TimeStep                   1.0       # default=0.5 (fs)
MD.Opt.criterion          1.0e-4        # default=1.0e-4 (Hartree/Bohr)
```

6.2 キーワード

各キーワードについて下記に説明していきます。この節では OpenMX のすべてのキーワードは網羅しておらず、また各キーワードについては対応する節で更に詳しく説明しています。

ファイル名

System.CurrrentDir
このキーワードにより、出力ファイルの出力場所を指定します。デフォルト値は「./」です。

System.Name
このキーワードで出力ファイルのファイル名を指定します。

DATA.PATH
VPS および PAO ディレクトリへのパスを下記のように指定します。

```
DATA.PATH    ../DFT_DATA13    # default=../DFT_DATA13
```

絶対パス、相対パスとも指定可能です。デフォルト値は「../DFT_DATA13」です。

level.of.stdout
このキーワードにより、標準出力への出力情報のレベルを制御します。「level.of.stdout=1」を指定した場合、最小限の情報が出力されます。「level.of.stdout=2」の場合、最小限の出力に加えて追加の情報が出力されます。「level.of.stdout=3」は開発者向けのオプションです。デフォルト値は 1 です。

level.of.fileout
このキーワードにより、出力ファイルへの出力情報のレベルを制御します。「level.of.fileout=0」を指定した場合、最小限の情報が出力されます（Gaussian cube およびグリッドファイルの出力無し）。「level.of.fileout=1」は標準的な出力情報レベルです。「level.of.fileout=2」の場合、標準の出力に加えて追加の情報が出力されます。デフォルト値は 1 です。

原子種の定義

Species.Number
このキーワードにより、系に含まれる原子種の数を指定します。

Definition.of.Atomic.Species
擬原子基底軌道と擬ポテンシャルの両方のファイル名を明記することにより、原子種を指定してください。これらのファイルはそれぞれ「DFT_DATA13/PAO」および「DFT_DATA13/VPS」ディレクトリに置かなければなりません。例えば以下のように指定します。

```
<Definition.of.Atomic.Species
 H   H5.0-s1>1p1>1      H_CA13
 C   C5.0-s1>1p1>1      C_CA13
Definition.of.Atomic.Species>
```

原子種の指定は「<Definition.of.Atomic.Species」で始め、「Definition.of.Atomic.Species>」で終わらなければなりません。1 番目の列では任意の原子種の名前を指定します。この名前は、後述する「Atoms.SpeciesAndCoordinates」キーワードにて原子座標を指定するのに使います。2 番目の列では擬原子基底軌道の拡張子無しのファイル名と、プリミティブ軌道の数および縮約された軌道の数を指定します。例では基底軌道は「H4.0-s1>1p1>1」となっていますが、H4.0 が「DFT_DATA13/PAO」ディレクトリにある擬原子基底軌道の拡張子無しファイル名を示し、s1>1 は、1 つの s 軌道のプリミティブ軌道から 1 つの最適化された軌道が作られていることを示します。つまり縮約は行わないことを意味します。縮約が無い場合、「s1>1」と記述する代わりに「s1」という簡単な記法を使用できます。つまり、「H4.0-s1p1」は「H4.0-s1>1p1>1」と同等ということです。3 番目の列では擬ポテンシャルの拡張無しファイル名を指定します。このファイルは「DFT_DATA13/VPS」ディレクトリに存在しなければなり

ません。同一元素に対して異なる基底軌道と擬ポテンシャルを割り当てることで、異なる原子種を定義することもできます。例えば複数の原子種を以下のように指定できます。

```
<Definition.of.Atomic.Species
 H1   H5.0-s1p1          H_CA13
 H2   H5.0-s2p2d1        H_CA13
 C1   C5.0-s2p2          C_CA13
 C2   C5.0-s2p2d2        C_CA13
Definition.of.Atomic.Species>
```

この柔軟性の高い記述法を活用し、系の特性を決定している領域に属する原子には高レベルの基底関数を割り当て、不活性な領域に属する原子には低レベルの基底関数を割り当てることも可能であり、計算量の削減には有効な手段です。

原子

Atoms.Number
系の合計原子数を「Atoms.Number」キーワードで指定します。

Atoms.SpeciesAndCoordinates.Unit
原子座標の単位を「Atoms.SpeciesAndCoordinates.Unit」キーワードで指定します。オングストロームの場合は「Ang」、原子単位の場合は「AU」を指定します。「FRAC」と記述することで、規格化座標を指定することも可能です。規格化座標の場合には、後述する「Atoms.UnitVectors」のキーワードで与えられた **a**、**b**、**c** ベクトルを基底して、座標を指定します。その際には-0.5〜0.5 の範囲で座標が設定可能で、この範囲を超える座標は入力ファイルの読み込み後に自動的に調整されます。

Atoms.SpeciesAndCoordinates
原子座標および各スピン毎の電子数を「Atoms.SpeciesAndCoordinates」キーワードで以下の例のように指定します。

```
<Atoms.SpeciesAndCoordinates
  1   C      0.000000    0.000000    0.000000     2.0  2.0
  2   H     -0.889981   -0.629312    0.000000     0.5  0.5
  3   H      0.000000    0.629312   -0.889981     0.5  0.5
  4   H      0.000000    0.629312    0.889981     0.5  0.5
  5   H      0.889981   -0.629312    0.000000     0.5  0.5
Atoms.SpeciesAndCoordinates>
```

記述は「<Atoms.SpeciesAndCoordinates」で始まり、「Atoms.SpeciesAndCoordinates>」で終わります。1 列目には原子を特定する連番を記述します。2 列目には事前に「Definition.of.Atomic.Species」の

１列目で定義した原子種を指定します。3〜5 列目には、それぞれ x、y、z 座標を指定します。「Atoms.SpeciesAndCoordinates.Unit」キーワードで「FRAC」を指定している場合は 3〜5 列目はそれぞれ **a**、**b**、**c** 軸の規格化座標になり、値は-0.5〜0.5 の範囲で指定します。この範囲を外れる数値は入力ファイルが読み込まれる段階で自動的に調整されます。6 および 7 列目には、各原子のアップスピンとダウンスピン状態のそれぞれの初期電子数を設定します。アップスピン電子数とダウンスピン電子数の合計は原子の価電子の数に等しくなるようにします。「LSDA-CA」、「LSDA-PW」、あるいは「GGA-PBE」を使用してスピン分極系を計算する場合、6 および 7 列目の指定によって各原子に初期スピンを指定することができます。これにより強磁性状態や反強磁性状態などの計算が可能です。また SCF の収束を加速するためには、この初期スピン配置は基底状態に近い配置からスタートすることが重要です。

Atoms.UnitVectors.Unit
単位胞 (ユニットセル) におけるベクトルの単位を「Atoms.UnitVectors.Unit」キーワードで指定します。オングストロームの場合は「Ang」、原子単位の場合は「AU」を指定します。

Atoms.UnitVectors
単位胞のベクトル **a**、**b**、**c** を Atoms.UnitVectors キーワードで以下の様に指定します。

```
<Atoms.UnitVectors
  10.0   0.0   0.0
   0.0  10.0   0.0
   0.0   0.0  10.0
Atoms.UnitVectors>
```

記述は「<Atoms.UnitVectors」で始まり、「Atoms.UnitVectors>」で終わります。第 1〜3 行はそれぞれ単位胞のベクトル **a**、**b**、**c** に対応します。クラスタ計算の際にこのキーワードが指定されない場合、周期的に配置された孤立系の間で基底関数が重ならないように、単位胞の大きさが自動的に決定されます。詳細に関しては「セルサイズの自動決定」の章もご参照ください。

SCF または電子系

scf.XcType
交換相関ポテンシャルを「scf.XcType」キーワードで指定します。OpenMX Ver. 3.7 においては「LDA」、「LSDA-CA」、「LSDA-PW」、「GGA-PBE」から選択できます。ここで「LSDA-CA」は Ceperley-Alder [2] の局所スピン密度関数、「LSDA-PW」は 、その GGA 形式において密度勾配をゼロとした Perdew-Wang 局所スピン密度関数 [4] です。「GGA-PBE」は Perdew らが提案する GGA 汎関数 [5] です。

scf.SpinPolarization
電子系の非スピン分極あるいはスピン分極は「scf.SpinPolarization」キーワードで指定します。スピン分極の計算を行う場合は「ON」を指定し、非スピン分極の計算を行う場合は「OFF」を指定します。「scf.XcType」キーワードで「LDA」を使用する場合は「scf.SpinPolarization」を「OFF」に設定しなければなりません。前述の 2 つのオプションの他、ノンコリニア DFT 計算を行う場合には「NC」とい

うオプションを指定して下さい。この計算については「ノンコリニア DFT」の節もご参照ください。

scf.partialCoreCorrection
交換相関エネルギーおよびそのポテンシャルの計算において部分内殻補正（Partial Core Correction、PCC）を行う場合には、「scf.partialCoreCorrection」キーワードを指定します。これを「ON」に指定すると PCC を行い、「OFF」の場合には PCC を行いません。通常は、このフラグは「ON」にするべきです。何故ならば、PCC 擬ポテンシャルを使用する場合には PCC を行うべきであり、また非 PCC 擬ポテンシャルの場合でも PCC 電荷がゼロなので結果には影響を及ぼさないからです。デフォルトは「ON」です。

scf.Hubbard.U
LDA+U および GGA+U 計算の場合、「scf.Hubbard.U」キーワードを「ON」に設定してください (ON|OFF)。デフォルトでは「OFF」に設定されています。

scf.Hubbard.Occupation
LDA+U 法では、「scf.Hubbard.Occupation」キーワードを指定することで、「onsite」、「full」および「dual」の 3 つの占有数演算子から選択することができます。

Hubbard.U.values
下記のキーワードにより各原子種の軌道に対する実効的な U の値を定義します。

```
<Hubbard.U.values                  #  eV
 Ni  1s 0.0 2s 0.0 1p 0.0 2p 0.0 1d 4.0 2d 0.0
 O   1s 0.0 2s 0.0 1p 0.0 2p 0.0 1d 0.0
Hubbard.U.values>
```

記述は「<Hubbard.U.values」で始まり、「Hubbard.U.values>」で終わります。
「Definition.of.Atomic.Species」で指定した全ての基底軌道に対して、上記の形式にて実効的な U の値を設定しなければなりません。「1s」や「2s」はそれぞれ 1s および 2s 軌道を示し、「1s」の後にくる数字が 1s 軌道に対する実効的な U 値（eV）です。同様の規則が、p および d 軌道にも当てはまります。

scf.Constraint.NC.Spin
スピン磁気モーメントのノンコリニア方位を制御する制約条件付 DFT 計算を行う場合には「scf.Constraint.NC.Spin」キーワードを「ON」に設定してください。このキーワードは「ON」か「OFF」に設定することができます。

scf.Constraint.NC.Spin.v
スピン磁気モーメントのノンコリニア方位を制御する制約条件付 DFT 計算を行う場合には、「scf.Constraint.NC.Spin.v キーワードによってペナルティ関数の前因子（eV）を指定します。

scf.ElectronicTemperature
「scf.ElectronicTemperature」キーワードにて電子温度（K）を設定します。デフォルトでは 300（K）に設定されています。

scf.energycutoff

「scf.energycutoff」キーワードで積分グリッド間隔を定義するカットオフエネルギーを指定します。この積分グリッドは差電子クーロンポテンシャルと交換相関ポテンシャルに対する行列要素の計算、および高速フーリエ変換（FFT）を用いた Poisson 方程式の解法のために使用されます。デフォルトでは 150（Ryd）に設定されています。

scf.Ngrid

「scf.Ngrid」キーワードによって **a**、**b**、**c** 軸を離散化させる際のグリッド数を指定します。
通常は「scf.energycutoff」で指定した値を使って実空間の離散化を行いますが、「scf.Ngrid」でグリッド数を指定した場合には、「scf.energycutoff」ではなくこちらが優先されます。

scf.maxIter

SCF の最大反復回数を「scf.maxIter」キーワードで設定します。SCF 反復ループは、収束条件が満たされない場合でもこのキーワードで指定した回数で終了します。デフォルト値は 40 です。

scf.EigenvalueSolver

固有値問題の計算手法を「scf.EigenvalueSolver」で指定します。O(N) 分割統治法は「DC」、O(N) クリロフ部分空間法は「Krylov」、数値厳密な低次スケーリング法は「ON2」、クラスタ計算は「Cluster」、バンド計算は「Band」を指定します。

scf.Kgrid

「scf.EigenvalueSolver」キーワードにて「Band」を指定する場合、k 空間の第 1 ブリルアンゾーンを離散化するための格子の数（n1, n2, n3）を「scf.Kgrid」キーワードで指定しなければなりません。k 空間の逆格子ベクトル $\tilde{\mathbf{a}}$、$\tilde{\mathbf{b}}$、$\tilde{\mathbf{c}}$ を離散化するために「n1 n2 n3」のように指定してください。OpenMX では、Monkhorst-Pack 法 [25] にしたがって離散化された k 点を生成します。

scf.ProExpn.VNA

中性原子ポテンシャル VNA を射影演算子で展開する場合 [29] には、「scf.ProExpn.VNA」キーワードを「ON」に設定し、それ以外の場合は「OFF」を設定してください。デフォルトでは「ON」になっています。

```
scf.ProExpn.VNA      ON      # ON|OFF, default = ON
```

「scf.ProExpn.VNA=OFF」と設定されている場合、VNA ポテンシャルの行列要素は実空間の離散グリッドを用いて計算されます。

scf.Mixing.Type

「scf.Mixing.Type」キーワードによって、SCF 計算の次の反復ステップに入力される電子密度 (もしくは密度行列) を生成するための電子密度混合法を指定します。単純混合法（「Simple」）、Guaranteed-Reduction Pulay 法 [39]（「GR-Pulay」）、RMM-DIIS 法 [40]（「RM-DIIS」）、Kerker 法 [41]（「Kerker」）、あるいは RMM-DIISK 法 [40]（「RM-DIISK」）のいずれかを指定することができます。OpenMX の単純混合法は収束の速度を上げるために収束履歴を参照するように改良されています。「GR-Pulay」、「RMM-DIIS」、「Kerker」あるいは「RMM-DIISK」を使用する際には、以下の点に注意することで SCF 計算

の高速化が可能です。

- Pulay 法に準拠する方法で混合を開始する前にある程度の収束を必要です。そのため、やや大きめの「scf.Mixing.StartPulay」の値を使用します。「scf.Mixing.StartPulay」の適切な値は 10〜30 です。
- 金属系の場合は高い「scf.ElectronicTemperature」の値を使用します。「scf.ElectronicTemperature」が小さいと数値的に不安定な挙動がよく見られます。
- 「scf.Mixing.History」の値を大きくします。ほとんどの場合、「scf.Mixing.History=30」が妥当な値です。

また前述した混合法のうち、一番ロバスト性が高いのは「RMM-DIISK」です。

scf.Init.Mixing.Weight
単純法、GR-Pulay 法, RMM-DIIS 法、Kerker 法、および RMM-DIISK 法で使用する初期の混合比を「scf.Init.Mixing.Weight」キーワードで指定します。有効な範囲は 0 <scf.Init.Mixing.Weight< 1 で、デフォルト値は 0.3 です。

scf.Min.Mixing.Weight
単純および Kerker 混合法における混合比の下限を「scf.Min.Mixing.Weight」で指定します。デフォルトでは 0.001 に設定されています。

scf.Max.Mixing.Weight
単純および Kerker 混合法における混合比の上限を「scf.Max.Mixing.Weight」で指定します。デフォルトでは 0.4 に設定されています。

scf.Kerker.factor
Kerker および RMM-DIISK 混合法における Kerker 因子を指定します。このキーワードの指定が無い場合、適切な値が自動的に設定されます。詳しくは「SCF 収束」の節を参照してください。

scf.Mixing.History
GR-Pulay 法 [39]、RMM-DIIS 法 [40]、Kerker 法 [41]、および RMM- DIISK 法 [40] では、SCF の次の反復ステップにおける入力電子密度を、過去の SCF 反復の電子密度に基づき推定します。「scf.Mixing.History」キーワードは、この推定に使用する過去の SCF 反復ステップ数を指定します。例えば、「scf.Mixing.History」が 3 に設定されている場合、6 回目の SCF 反復では過去の第 5、4、3 ステップの電子密度が考慮されます。30 程度が適切な選択値です。

scf.Mixing.StartPulay
GR-Pulay 法、RMM-DIIS 法、Kerker 法、および RMM-DIISK 法を開始する SCF ステップを「scf.Mixing.StartPulay」で指定します。これらの方法を開始するまでの SCF ステップでは単純あるいは Kerker 混合法が使用されます。デフォルトでは 6 に設定されています。

scf.Mixing.EveryPulay
Pulay タイプの混合法における残差ベクトルは、SCF ステップが進むに従い、相互に線形従属になる傾

向にあり、このため収束が困難になります。線形従属を避ける方法のひとつは、Kerker 混合の合間に時々Pulay タイプの混合を挿入するというものです。この方法における Pulay タイプ混合の使用頻度を、「scf.Mixing.EveryPulay」キーワードで指定します。例えば「scf.Mixing.EveryPulay=5」と設定した場合、SCF 反復 5 回につき 1 回の Pulay 混合が実施され、その他の反復では Kerker 法が使用されます。「scf.Mixing.EveryPulay=1」は従来の Pulay タイプ混合と同じものになります。このキーワード「scf.Mixing.EveryPulay」は、「RMM-DIISK」でのみサポートされており、デフォルト値は 1 です。

scf.criterion
SCF 計算の収束条件を「scf.criterion」キーワードで指定します（Hartree 単位）。SCF 反復は、dUele<scf.criterion という条件が満たされると終了します。ここで dUele とは、現在の SCF 反復とひとつ前の反復とのバンドエネルギーの絶対差です。デフォルト値は 1.0e-6（Hartree 単位）です。

scf.Electric.Field
「scf.Electric.Field」キーワードによって、のこぎり波形で表される外部均一電場を導入することが可能です。例えば、**a** 軸に沿って 1.0 GV/m (10^9 V/m) の電場を適用するには、次のように指定します。

```
scf.Electric.Field   1.0 0.0 0.0   # default=0.0 0.0 0.0 (GV/m)
```

電場の符号は電子に作用するものとして与えられます。デフォルト値は「0.0 0.0 0.0」です。

scf.system.charge
電子および正孔のドーピングの量を「scf.system.charge」キーワードで指定します。プラス・マイナス符号はそれぞれ正孔と電子のドーピングを表します。デフォルト値は 0 です。

scf.SpinOrbit.Coupling
スピン軌道相互作用を有効にする場合にはこのキーワードを「ON」に設定し、それ以外では「OFF」に設定してください。スピン軌道相互作用を使用する場合、j 依存擬ポテンシャルを使用しなければなりません。これについては「相対論的効果」の節も参照して下さい。

1D FFT

1DFFT.EnergyCutoff
局在擬原子軌道と非局所ポテンシャルのプロジェクタの動径関数をフーリエ変換するエネルギー領域を「1DFFT.EnergyCutoff」キーワードで指定します。このエネルギー領域はフーリエ空間における動径方向の最大値となります。デフォルト値は 3600（Ryd）です。

1DFFT.NumGridK
0 から 1DFFT.EnergyCutoff までのエネルギー領域は「1DFFT.NumGridK」キーワードで指定されるグリット数で離散化されます。局在擬原子軌道および非局所ポテンシャルのプロジェクタの動径関数のフーリエ変換された値は、**k** 空間上の動径方向の関数として、0 から 1DFFT.EnergyCutoff までの範囲で、このグリット上で保持されています。デフォルト値は 900 です。

1DFFT.NumGridR

実空間上の動径グリット数を「1DFFT.NumGridR」キーワードで指定します。この値は局在擬原子軌道および非局所ポテンシャルのプロジェクタの動径関数をフーリエ変換する際の数値積分に使用されます。デフォルト値は 900 です。

軌道最適化

orbitalOpt.Method
軌道最適化の方法を「orbitalOpt.Method」キーワードで指定します。軌道最適化を行わない場合はこの値を「OFF」に設定してください。軌道最適化を行う場合は、次のオプションから選択します。「atoms」を指定した場合、各原子単位の基底関数が最適化されます。「species」を指定した場合、各原子種の基底関数が最適化されます。「atom」および「species」のどちらの場合でも、動径波動関数 R は磁気量子数には依存しないという条件のもとで最適化されます。これにより最適化基底関数を用いた計算において、全エネルギーの回転不変性が保証されます。「atom」を指定した場合には最適化された基底関数は化学的環境に依存するため、同一原子種であっても原子毎に異なります。一方、「species」を指定した場合には「Definition.of.Atomic.Species」で定義した同一名の原子種の基底関数は同一の軌道に最適化されます。ほぼ同様の化学的環境を持つ原子に対して同一の最適化基底を生成したい場合に、有用な方法です。

orbitalOpt.scf.maxIter
軌道最適化における SCF 反復の最大回数を「orbitalOpt.scf.maxIter」キーワードで指定します。

orbitalOpt.Opt.maxIter
軌道最適化の反復の最大回数を「orbitalOpt.Opt.maxIter」キーワードで指定します。軌道最適化の反復は、収束条件が達成しなかった場合でも、同キーワードで設定した回数で終了します。

orbitalOpt.Opt.Method
軌道最適化の収束方法として、2 つの手法がサポートされています。「EF」は固有ベクトル追跡法、「DIIS」は反復部分空間における直接反転法です。それぞれのアルゴリズムは構造最適化のそれと同じです。「orbitalOpt.Opt.Method」キーワードでは「EF」あるいは「DIIS」を指定してください。

orbitalOpt.StartPulay
「orbitalOpt.StartPulay」キーワードで指定した最適化ステップから、準ニュートン法である「EF」または「DIIS」法を開始します。

orbitalOpt.HistoryPulay
準ニュートン法である「EF」および「DIIS」法において、次ステップでの縮約係数を推定するために参照する過去のステップ数を「orbitalOpt.HistoryPulay」キーワードで指定します。

orbitalOpt.SD.step
準ニュートン法である「EF」および「DIIS」法を開始するまでの最適化ステップは最急降下法が適用されます。最急降下法で使用する前因子は「orbitalOpt.SD.step」キーワードで指定します。多くのケースにおいて、「orbitalOpt.SD.step」の適切な値は 0.001 程度となります。

orbitalOpt.criterion

軌道最適化の収束条件（(Hartree/bohr)^2）を「orbitalOpt.criterion」キーワードで指定します。「微分のノルム <orbitalOpt.criterion」という条件が満たされた時に反復ループが終了します。

CntOrb.fileout
最適化動径関数をファイルに出力したい場合は、「CntOrb.fileout」キーワードを「ON」にする必要があります。

Num.CntOrb.Atoms
最適化動径関数をファイルに出力する際の原子数を「Num.CntOrb.Atoms」キーワードで指定します。

Atoms.Cont.Orbitals
「Atoms.SpeciesAndCoordinates」キーワードの第 1 列で定義した原子の通し番号を用いて、最適化基底を出力する原子を「Atoms.Cont.Orbitals」キーワードで次のように指定します。

```
<Atoms.Cont.Orbitals
 1
 2
Atoms.Cont.Orbitals>
```

記述は「<Atoms.Cont.Orbitals」で始め、「Atoms.Cont.Orbitals>」で終わります。行の数は、「Atoms.Cont.Orbitals」で記述する数字と整合性がなければなりません。例えば、最適化擬原子軌道が「C_1.pao」と「H_2.pao」などとして保存された場合、その原子種名はキーワード「Definition.of.Atomic.Species」の設定における第一列の記号に対応し、記号の後の数字はキーワード「Atoms.SpeciesAndCoordinates」の設定における第一列の数値を意味します。これらの出力ファイル「C_1.pao」と「H_2.pao」は基底関数の入力データとして使用可能です。

O(N)SCF 計算

orderN.HoppingRanges
各原子を中心とする球の半径 (Å) を「orderN.HoppingRanges」キーワードで定義します。DC 法および O(N) クリロフ部分空間法では、この球内に含まれる原子を選択することで切り取られたクラスタが構成されます。

orderN.KrylovH.order
切り取られた各クラスタのハミルトニアンに対し、クリロフ部分空間の次元を「orderN.KrylovH.order」キーワードで指定します。

orderN.KrylovS.order
後述のキーワードで「orderN.Exact.Inverse.S=off」に設定した場合、重なり積分の逆行列はクリロフ部分空間法を用いて近似されます。この時、切り取られた各クラスタに対する重なり行列のクリロフ部分空間法の次元を「orderN.KrylovS.order」キーワードで指定します。デフォルト値は、「orderN.KrylovH.order」で設定した値の 4 倍です。

orderN.Exact.Inverse.S
キーワード「orderN.Exact.Inverse.S」を「on」に設定すると、切り取られた各クラスタの重なり行列の逆行列は厳密に評価されます。off に設定する場合は前述のキーワード「orderN.KrylovS.order」の項を参照してください。デフォルトでは「on」に設定されています。

orderN.Recalc.Buffer
キーワード「orderN.Recalc.Buffer」を「on」に設定すると、バッファ行列は各 SCF 反復毎に再計算されます。「off」の場合にはバッファ行列は初回の SCF ステップで計算され、その後の SCF 反復では固定されます。デフォルトでは「on」に設定されています。

orderN.Expand.Core
キーワード「orderN.Expand.Core」を「on」に設定すると、コア領域は半径 $1.2 \times r_{\min}$ の球の中にある原子から構成されます。ここで $r_{\min}$ は、中心原子と最隣接原子間の距離です。このコア領域はクリロフ部分空間を生成するときの第 1 ステップで使用されるベクトル群を定義します。「orderN.Expand.Core」が「off」の場合、中心の原子がコア領域と見なされます。デフォルトでは「on」に設定されています。

MD(分子動力学）あるいは構造最適化

MD.Type
分子動力学計算あるいは構造最適化のタイプを指定します。現在使用できるオプションは以下の通りです。MD 無し（「Nomd」)、NVE アンサンブル MD（「NVE」)、速度スケーリング法による NVT アンサンブル MD（「NVT_VS」）[17]、Nose-Hoover 法による NVT アンサンブル MD（「NVT_NH」）[18]、複数熱浴付き MD（「NVT_VS2」あるいは「NVT_VS4」)、最急降下法による構造最適化（「SD」)、DIIS 最適化法（「DIIS」)、固有値ベクトル追跡法（「EF」）[45]、および有理関数法（「RF」）[46]。詳細は「構造最適化」および「分子動力学」の節を参照してください。

MD.Fixed.XYZ
次のキーワードを指定することで、構造最適化および分子動力学シミュレーションの際に任意の原子座標を初期座標に固定することができます。各原子の x、y、z 座標に対して、以下のように設定します。

```
<MD.Fixed.XYZ
  1  1 1 1
  2  1 0 0
MD.Fixed.XYZ>
```

上記の例は 2 つの原子から構成される系の場合です。系に N 個の原子がある場合、N 個分の指定が必要です。第 1 列は「Atoms.SpeciesAndCoordinates」で個々の原子を特定するのに使用した連番です。第 2〜4 列は、それぞれ x、y、z 座標のフラグです。フラグが「1」だと座標は固定、「0」だと緩和されます。上記の例では、原子 1 の x、y、z 座標は固定、原子 2 は x 座標のみ固定されています。デフォルトでは全ての座標が緩和される指定になっています。このキーワードによる原子座標の固定は、すべての構造最適化および分子動力学の方法において有効です。

MD.maxIter
MD 計算や構造最適化計算における最大の反復回数を「MD.maxIter」キーワードで指定します。

MD.TimeStep
時間ステップ（fs）を「MD.TimeStep」キーワードで指定します。

MD.Opt.criterion
「MD.Type」キーワードによって構造最適化の方法を選択している場合には、「MD.Opt.criterion」キーワードでその収束条件（Hartree/Bohr）を設定します。原子にかかる力の最大絶対値が「MD.Opt.criterion」で指定した値より小さくなった場合に、構造最適化は終了します。

MD.Opt.DIIS.History
「DIIS」、「EF」、「RF」による構造最適化を行う場合、「MD.Opt.DIIS.History」キーワードは構造最適化のために参照する過去のステップ数を指定します。デフォルト値は 3 です。

MD.Opt.StartDIIS
「DIIS」、「EF」、「RF」による構造最適化を開始するステップを「MD.Opt.StartDIIS」キーワードで指定します。DIIS タイプの構造最適化を開始する以前のステップでは最急降下法が使用されます。デフォルト値は 5 です。

MD.TempControl
キーワード「MD.TempControl」により MD および NVT アンサンブルにおける原子運動の温度を指定します。「NVT_VS」を選択した場合、原子運動の温度を下記の例のように制御できます。

```
<MD.TempControl
  3
  100   2  1000.0  0.0
  400  10   700.0  0.4
  700  40   500.0  0.7
MD.TempControl>
```

記述は「<MD.TempControl」で開始し、「MD.TempControl>」で終わります。最初の数字「3」は、続く温度指定の行数を指します。例では 3 行あります。後続する行の第 1 列は MD ステップ数を指し、第 2 列は速度スケーリングを行う MD ステップの間隔を指定します。例では、100 ステップ目までは 2 ステップ毎に、100〜400 ステップ間は 10 ステップ毎に、400〜700 ステップ間は 40 ステップ毎に速度スケーリングを行います。第 3、4 列はそれぞれ温度（K）とスケーリングパラメータ α を指定します。詳細は「分子動力学」の節を参照してください。一方、NVT_NH の場合、以下の記述で原子運動の温度を制御できます。

```
<MD.TempControl
  4
  1    1000.0
  100  1000.0
```

```
  400   700.0
  700   600.0
MD.TempControl>
```

記述は「<MD.TempControl」で開始し、「MD.TempControl>」で終わります。最初の数字「4」は、続く温度指定の行数を指します。この例では 4 行あります。後続する行の第 1 、2 列は、それぞれ MD ステップ数と原子運動の温度を指定します。指定された MD ステップ間の温度は線形補完されます。

NH.Mass.HeatBath
キーワード「MD.Type」によって「NVT_NH」を選択した場合、このキーワードで熱浴の質量を設定します。デフォルト値は 20 です。単位は統一原子質量単位です（炭素原子の主同位体の質量を 12.0 とする単位)。

MD.Init.Velocity
分子動力学のシミュレーションでは、下記の記述により各原子に初期速度を与えることができます。

```
<MD.Init.Velocity
 1    3000.000  0.0  0.0
 2   -3000.000  0.0  0.0
MD.Init.Velocity>
```

これは 2 つの原子からなる系の例です。N 個の原子がある場合、同様に N 行の記述が必要です。第 1 列は「Atoms.SpeciesAndCoordinates」キーワードで指定した原子の連番と同じです。第 2、3、4 列は、各原子の速度の x、y、z 成分で、単位は m/s です。「MD.Init.Velocity」キーワードは、「MD.Fixed.XYZ」キーワードと共存できます。

バンド分散

Band.dispersion
バンド分散を評価する場合は「Band.dispersion」キーワードを「ON」に設定してください。

Band.KPath.UnitCell
「Band.KPath.UnitCell」キーワードによって、下記の例のようにバンド分散の計算で使用する単位胞ベクトルを指定します。

```
<Band.KPath.UnitCell
 3.56 0.0 0.0
 0.0 3.56 0.0
 0.0 0.0 3.56
Band.KPath.UnitCell>
```

記述は「<Band.KPath.UnitCell」で開始し、「Band.KPath.UnitCell>」で終了します。
「Band.KPath.UnitCell」キーワードの記述がある場合、同キーワードで指定した単位胞ベクトルからバンド分散の計算に使用する逆格子ベクトルを算出します。「Band.KPath.UnitCell」を指定しない場合には、「Atoms.UnitVectors」キーワードで指定した単位胞ベクトルから計算した逆格子ベクトルが使用されます。面心立方格子構造（fcc）や体心立方格子構造（bcc）の場合には「Band.KPath.UnitCell」キーワードで逆格子ベクトルを再定義した方が、特殊 **k** 点の指定が容易です。

Band.Nkpath
バンド分散の計算における経路の数を「Band.Nkpath」で指定します。

Band.kpath
バンド分散の経路を「Band.kpath」キーワードで下記のように指定します。

```
<Band.kpath
  15  0.0 0.0 0.0   1.0 0.0 0.0   g X
  15  1.0 0.0 0.0   1.0 0.5 0.0   X W
  15  1.0 0.5 0.0   0.5 0.5 0.5   W L
  15  0.5 0.5 0.5   0.0 0.0 0.0   L g
  15  0.0 0.0 0.0   1.0 1.0 0.0   g X
Band.kpath>
```

記述は「<Band.kpath」で始まり、「Band.kpath>」で終わります。行数は「Band.Nkpath」で指定した数と同じでなくてはなりません。第 1 列には経路上で固有値を計算する格子の数を指定します。続く (n1, n2, n3) および (n1', n2', n3') は、逆格子ベクトルを基底とする第 1 ブリルアンゾーンの経路の開始および終了の **k** 点を指定します。「Band.KPath.UnitCell」キーワードを指定している場合、バンド分散を計算するための逆格子ベクトルは「Band.KPath.UnitCell」で設定した単位胞ベクトルを用いて計算されます。「Band.KPath.UnitCell」の定義が無い場合は「Atoms.UnitVectors」で指定した単位胞ベクトルを用いて逆格子ベクトルを計算します。最後の 2 つの英文字は経路の開始および終了の **k** 点の名前を指定します。

再スタート

scf.restart
過去の計算で生成されたリスタートファイル「*_rst/*」を用いて SCF 計算を再スタートさせる場合には、このキーワード「scf.restart」を「ON」に設定してください。

分子軌道（MO）の出力

MO.fileout
分子軌道をファイルに出力したい場合は、キーワード「MO.fileout」を「ON」に設定してください。

num.HOMOs
出力する最高被占分子軌道（HOMO）の数を「num.HOMOs」キーワードで指定します。

num.LUMOs
出力する最低空分子軌道（LUMOS）の数を「num.LUMOs」キーワードで指定します。

MO.Nkpoint
「MO.fileout=ON」および「scf.EigenvalueSolver=Band」を設定している場合、「MO.Nkpoint」キーワードで MO を出力する **k** 点の数を指定します。

MO.kpoint
MO を出力する **k** 点を「MO.kpoint」キーワードで以下のように指定します。

```
<MO.kpoint
  0.0  0.0  0.0
MO.kpoint>
```

記述は「<MO.kpoint」で開始し、「MO.kpoint>」で終わります。k 点は逆格子ベクトルを基底して (n1, n2, n3) の 3 つの数字で指定します。ここで、逆格子ベクトルは「Band.kpath」の場合と同様な方法で生成されたものが使用されます。

DOS と PDOS

Dos.fileout
全状態密度（DOS）および射影した部分状態密度（PDOS）を評価する場合は「Dos.fileout」キーワードを「ON」に設定してください。

Dos.Erange
DOS 計算におけるエネルギー範囲を「Dos.Erange」キーワードで下記のように指定します。

```
Dos.Erange                -10.0  10.0
```

ここで 2 つの値はそれぞれエネルギー範囲の下限と上限（eV）です。

Dos.Kgrid
DOS 計算を行う上で、第 1 ブルリアンゾーンを離散化するために (n1, n2, n3) の格子点を「Dos.Kgrid」キーワードで指定します。

開発者向けインタフェース

HS.fileout
後処理計算において、Kohn-Sham ハミルトニアン行列、重なり行列、密度行列を利用する場合には

「HS.fileout」キーワードを「ON」に設定してください。これによりバイナリ形式のデータが「*.scfout」に格納されます。ここで、「*」はキーワード「System.Name」で設定したファイル名です。このデータの利用に関しては「開発者向けインタフェース」の章で詳しく説明します。

Voronoi 電荷

Voronoi.charge
Voronoi 電荷を計算する場合には「Voronoi.charge」キーワードを「ON」に設定してください。計算結果は「*.out」ファイルに保存されます。ここで「*」は「System.Name」で設定したファイル名です。

7　出力ファイル

「level.of.fileout=0」と設定された場合、以下のファイルが生成されます。ここで、「*」はキーワード「System.Name」で設定されたファイル名です。

- *.out

  SCF 計算の履歴、構造最適化の履歴、Mulliken 電荷、全エネルギー、および双極子モーメントが保存されます。

- *.xyz

  MD または構造最適化により得られた最終的な幾何学的構造が保存されます。このファイルは xmakemol や XCrySDen で可視化できます。

- *.bulk.xyz

  「scf.EigenvalueSolver=Band」の場合、コピーされたセルの原子を含む原子座標が出力されます。このファイルは xmakemol や XCrySDen で可視化できます。

- *_rst/

  再スタートを行うための一連のファイルを保管しているディレクトリです。

- *.md

  各 MD ステップごとの原子座標が保存されます。このファイルは xmakemol や XCrySDen で可視化できます。

- *.md2

  最終 MD ステップにおける原子座標が保存されます。指定した原子種記号を用いて原子が指定されています。

- *.cif

  Material Studio で可視化できる cif 形式の初期原子座標が保存されます。

- *.ene

  MD ステップごとの計算値が保存されます。保存されている各数値の内容は「iterout.c」 ルーチン中で確認できます。

「level.of.fileout=1」と指定した場合、「level.of.fileout=0」で生成されるファイルに加え、以下の Gaussian cube 形式ファイルが生成されます。ここで、「*」はキーワード「System.Name」により指定されたファイル名です。

- *.tden.cube

  Gaussian cube 形式の全電子密度が保存されています。

- *.sden.cube

  「LSDA-CA」、「LSDA-PW」、「GGA-PBE」を使用したスピン分極計算が実行される場合に、スピン電子密度が保存されています。

- *.dden.cube

  構成原子の孤立原子状態の電子密度の重ね合わせを基準として計算された差電子密度が保存されています。

- *.v0.cube

  アップスピンの非局所ポテンシャルを除く Kohn-Sham ポテンシャルが保存されています。

- *.v1.cube

  スピン分極計算における、ダウンスピンの非局所ポテンシャルを除く Kohn-Sham ポテンシャルが保存されています。

- *.vhart.cube

  差電子密度から生じる差電子 Hartree ポテンシャルが保存されています。

「level.of.fileout=2」の場合、「level.of.fileout=1」の場合に生成されるファイルに加え、以下のファイルが生成されます。ここで、「*」はキーワード「System.Name」で指定されたファイル名です。

- *.vxc0.cube

  Gaussian cube 形式のアップスピンの交換相関ポテンシャルが保存されています。

- *.vxc1.cube

  Gaussian cube 形式のダウンスピンの交換相関ポテンシャルが保存されています。

- *.grid

  数値積分およびポアソン方程式の解に使用される実空間格子が保存されています。

「MO.fileout=ON」および「scf.EigenvalueSolver=Cluster」の場合、以下のファイルも生成されます。

- *.homo0_0.cube, *.homo0_1.cube, ...

  Gaussian cube 形式で最高占有軌道 (HOMO) から順番に占有軌道を出力したファイルです。「homo」に続く最初の数字はスピン状態を意味します (up=0, down=1)。また 2 番目の数字は固有状態を表し、0、1、2 はそれぞれ HOMO、HOMO-1、HOMO-2 に対応します。

- *.lumo0_0.cube, *.lumo0_1.cube, ...

  Gaussian cube 形式で最低非占有軌道 (LUMO) から順番に非占有軌道を出力したファイルです。「lumo」に続く最初の数字はスピン状態を意味します (up=0, down=1)。また 2 番目の数字は固有状態を表し、0、1、2 はそれぞれ LUMO、LUMO+1、LUMO+2 に対応します。

「MO.fileout=ON」および「scf.EigenvalueSolver=Band」の場合、以下のファイルも生成されます。

- *.homo0_0_0_r.cube, *.homo1_0_1_r.cube, ... *.homo0_0_0_i.cube, *.homo1_0_1_i.cube, ...

  Gaussian cube 形式で最高占有軌道 (HOMO) から順番に占有軌道を出力したファイルです。「homo」に続く最初の数字は、キーワード「MO.kpoint」により指定される k 点の番号を意味します。また 2 番目の数字はスピン状態です (up=0, down=1)。3 番目の数字は固有状態を表し、0、1、2 はそれぞれ HOMO、HOMO-1、HOMO-2 に対応します。「r」および「i」は波動関数の実数および虚数部を意味します。

- *.lumo0_0_0_r.cube, *.lumo1_0_1_r.cube, ... *.lumo0_0_0_i.cube, *.lumo1_0_1_i.cube, ...

  Gaussian cube 形式で最低非占有軌道 (LUMO) から順番に非占有軌道を出力したファイルです。「lumo」 に続く最初の数字は、キーワード「MO.kpoint」により指定される k 点の番号を意味します。また 2 番目の数字はスピン状態です (up=0, down=1)。3 番目の数字は固有状態を表し、0、1、2 はそれぞれ LUMO、LUMO+1、LUMO+2 に対応します。「r」および「i」は波動関数の実数および虚数部を意味します。

「Band.Nkpath」が 0 以上で、かつ「scf.EigenvalueSolver=Band」の場合、以下のファイルも生成されます。

- *.Band

  バンド分散のデータファイルが保存されています。

「Dos.fileout=ON」の時、以下のファイルも生成されます。

- *.Dos.val

  状態密度を計算するための固有値のデータファイル。

- *.Dos.vec

  状態密度を計算するための固有ベクトルのデータファイル。

「scf.SpinPolarization=NC」および「level.of.fileout=1 もしくは 2」の場合、以下のファイルも生成されます。

- *.nco.xsf

  Mulliken 解析により各原子に射影されたノンコリニア軌道磁気モーメントが保存されたベクトルファイル。XCrySDen の「Display→Forces」を用いて可視化が可能です。

- *.nc.xsf

  Mulliken 解析により各原子に射影されたノンコリニアスピン磁気モーメントが保存されたベクトルファイル。XCrySDen の「Display→Forces」を用いて可視化が可能です。

- *.ncsden.xsf

  実空間格子上のノンコリニアスピン磁気モーメントが保存されたベクトルファイル。
  XCrySDen の「Display→Forces」を用いて可視化が可能です。

8 汎関数

OpenMX において、交換相関汎関数は局所密度近似 (LDA, LSDA) [2, 3, 4] もしくは一般化勾配近似 (GGA) [5] と呼ばれる近似手法によって取り扱われます。キーワード「scf.XcType」で、交換相関汎関数に対する近似の一つが選択できます。

```
scf.XcType                  LDA         # LDA|LSDA-CA|LSDA-PW|GGA-PBE
```

OpenMX Ver. 3.7 では、「LDA」、「LSDA-CA」、「LSDA-PW」、「GGA-PBE」が使用可能です。ここで、「LSDA-CA」は Ceperley-Alder の局所スピン密度汎関数であり [2]、「LSDA-PW」は Perdew-Wang による GGA の表現形式で密度勾配がゼロに設定された局所スピン密度汎関数です [4]。「GGA-PBE」は Perdew、 Burke、Ernzerhof により提案された GGA です [5]。この GGA は実空間での一次有限差分法を用いて実装されています。さらに、LDA+U (または GGA+U) 汎関数も使用可能です。詳細は「LDA+U」の章を参照してください。スピン (非) 分極およびノンコリニア計算を指定する関連キーワードは「scf.SpinPolarization」です。

```
scf.SpinPolarization        off         # On|Off|NC
```

スピン分極計算を実行する場合、「ON」に設定してください。非スピン分極計算を実行する場合には「OFF」に設定してください。キーワード「scf.XcType」に対して「LDA」を使用する場合には、キーワード「scf.SpinPolarization」は OFF にしてください。これらのオプションの他に、ノンコリニア DFT 計算を実行する場合は「NC」に設定して下さい。この計算に関しては「ノンコリニア DFT」の章も参照してください。

9 基底関数

9.1 概要

OpenMX は一粒子 Kohn-Sham 波動関数を展開するために数値擬原子軌道 (PAOs)χ を基底関数として使用します。PAO 関数は動径関数 R と実球面調和関数 Y との積として次式で与えられます。

$$\chi(\mathbf{r}) = R(r)Y(\hat{\mathbf{r}}),$$

この動径関数 R は数値の表として与えられ、実空間のカットオフ半径内で有限な関数です。つまり関数 R は所定のカットオフ半径を超えるとゼロになります。ADPACK により計算された PAO 関数は**プリミティブ**関数と呼ばれます。このプリミティブ PAO 関数を出発点として、OpenMX の軌道最適化法を用いて**最適化された** PAO 関数を得ることが可能です [28]。これらの基底関数に関するデータはファイル拡張子「pao」を持ったファイルに保存されています。OpenMX の計算が実行される際には、これらのファイルに保存されている数値データが読み込まれています。任意の動径距離 r に対する PAO 関数の値は補間法により得られます。ファイル拡張子「pao」を持ったファイルは、例えば「DFT_DATA13/PAO」などのディレクトリに保存して下さい。このディレクトリは以下のキーワードにより指定します。ただしディレクトリ名中に「PAO」を含める必要はありません。

```
DATA.PATH    ../DFT_DATA13    # default=../DFT_DATA13
```

絶対および相対パスの指定が可能であり、デフォルトは「../DFT_DATA13」となっています。OpenMX の入力ファイルにおいて、基底関数は以下のように「Definition.of.Atomic.Species」キーワードにより指定されます。

```
<Definition.of.Atomic.Species
  H   H5.0-s2p1      H_PBE13
  C   C5.0-s2p1      C_PBE13
Definition.of.Atomic.Species>
```

ここで基底関数の略語記号、H5.0-s2p1 を導入します。H5.0 は PAO 関数の拡張子なしのファイル名を表します。このファイルはキーワード「DATA.PATH」により指定された「DFT_DATA13/PAO」などのディレクトリに存在しなければなりません。5.0 は PAO 関数のカットオフ半径を意味します。また s2p1 は指定されたファイル中での二つの動径関数と一つの動径関数が s-軌道、p-軌道にそれぞれ割り当てられることを意味します。この場合、合計 5 つの PAO 基底関数 (2×1+1×3=5) が「H」に対して割り当てられます。

OpenMX の Web サイト (http://www.openmx-square.org/) では最適化基底関数のデータベース Ver. 2013 が無償公開されています。したがって一般のユーザーはこれらの最適化された基底関数を使用することが推奨されます。しかし上級者のために、プリミティブおよび最適化 PAO 関数について、続く節で説明します。

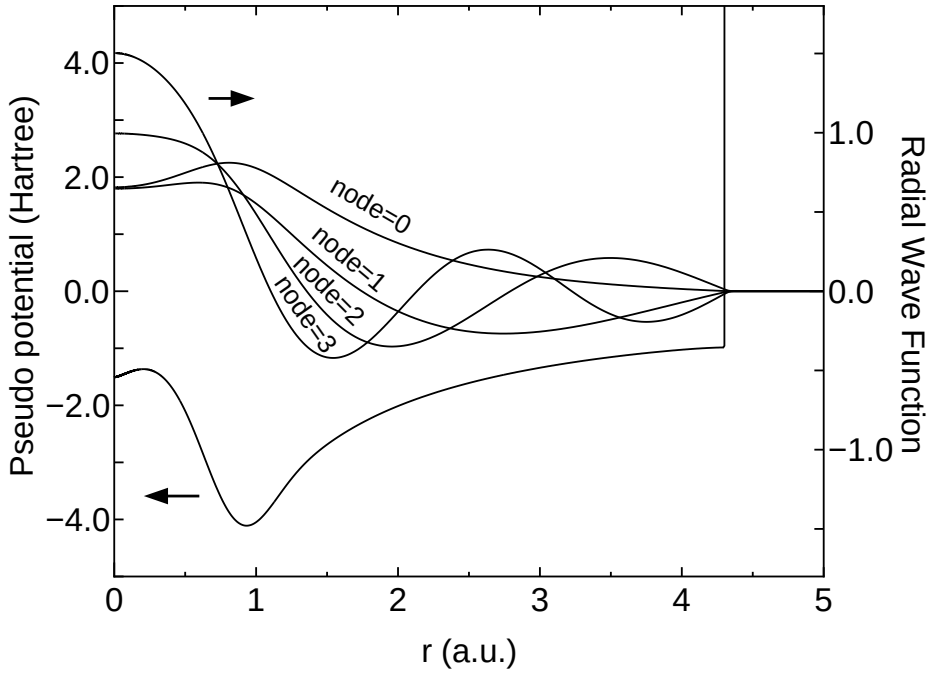


図 1: 閉じ込め擬ポテンシャル下で生成された炭素擬原子の s-軌道のプリミティブ動径関数。

9.2 プリミティブ基底関数

閉じ込め擬ポテンシャル下で擬原子の基底状態および励起状態を計算することで、プリミティブ動径関数が生成されます [28]。この計算は ADPACK を用いて実行可能です。一例として炭素擬原子の s-軌道のプリミティブ動径関数を図 1 に示します。これらの関数は数値の表として、ファイル拡張子「pao」を持つファイルに保存されています。図 1 から、基底状態はノードを持たず、第一励起状態は一つのノードを持っていることが分かります。より高い励起状態ではノード数は一つずつ増加しています。プリミティブ動径関数と実球面調和関数の積からプリミティブ基底関数が構成されます。さらに、このプリミティブ基底関数の線型結合によって一粒子 Kohn-Sham 波動関数が記述されます。計算の精度と効率は二つのパラメータ: カットオフ半径と基底関数の数により制御されます。一般的に図 2 に示すように、カットオフ半径と基底関数の数を増加させることで、収束解が得られます。しかしながら、大きなカットオフ半径を持ち、かつ多数の基底関数を使用する場合には、メモリー量や計算時間などの多大な計算リソースが必要となることに注意が必要です。各元素毎に適切なカットオフ半径と基底関数の数を決定する一般的な指針に関しては、参考文献 [28] で議論されています。この議論によれば、周期表の右側にある F や Cl などの元素に対して、十分な収束性を達成するためには高い角運動量をもつ基底関数が必要であり、周期表の左側にある Li や Na などの元素に対しては大きなカットオフ半径を使用すべきであるということが分かります。　OpenMX の Web サイト (http://www.openmx-square.org/) では最適化基底関数のデータベース Ver. 2013 が無償公開されています。したがって一般のユーザーはこれらの最適化された基底関数を使用することが推奨されます。

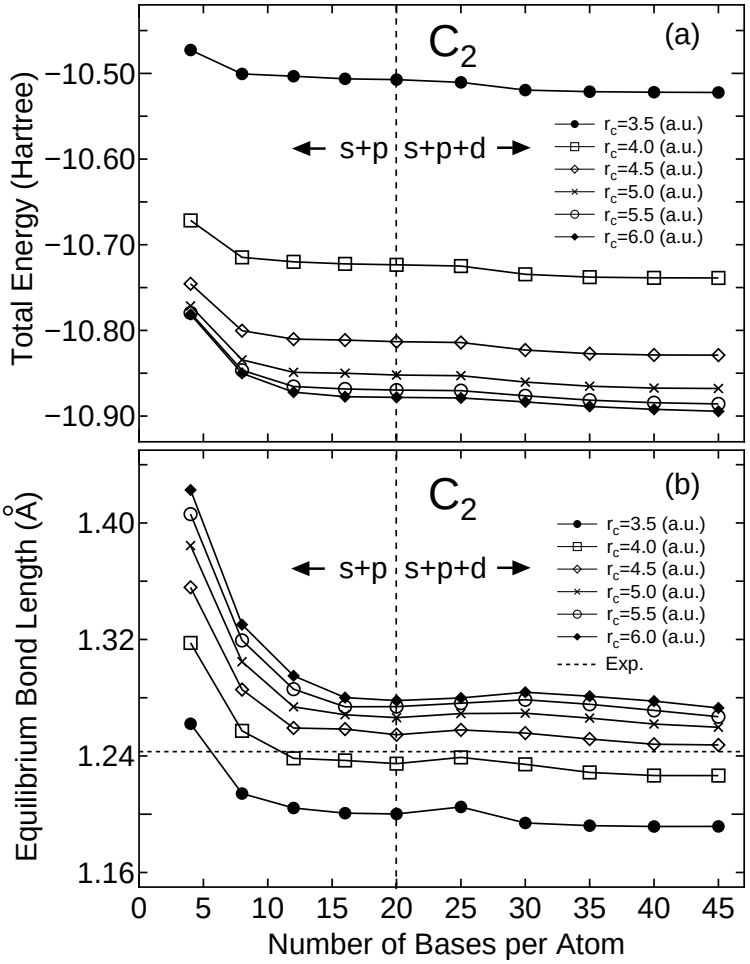


図 2: 炭素二量体の (a) 全エネルギーおよび (b) 平衡結合距離のカットオフ半径・基底関数の数に対する収束特性。

9.3 データベース Ver. 2013 により提供される最適化基底関数

最適化 PAO 関数はデータベース Ver. 2013 として、OpenMX の web サイト (http://www.openmx-square.org/) で一般公開されています。公開されている基底関数は軌道最適化法により生成され [28]、また一連のベンチマーク計算により十分にテストされていることから、一般ユーザーの最初の選択肢として、このデータベースの使用を推奨します。データベース Ver. 2013 内のほとんどの元素に対して、化学環境のトレーニングセットとして 3 種の系が選択され、選択された系に対して軌道最適化法により PAO 関数が変分原理に基づき最適化されています [28]。これらの最適化基底関数のセットは部分空間回転とグラム・シュミット直交化法の組み合わせ手法を用いて統合されており、一つの PAO ファイルに保存されています。最適化 PAO 関数はいくつかの異なる化学環境セットに対して最適化されているため、最適 PAO 関数の可搬性は高いと考えられます。実際、データベースのウェブサイト上にある一連のベンチマーク計算では、対応する全電子計算の結果とよく一致しています。これらベンチマーク計算を参考に、各元素に対しての適切な基底関数の選択 (カットオフ半径と基底関数の数) が可能でしょう。ベンチマーク計算に使われた入力ファイルもウェブサイトから入手することができますので、それらは

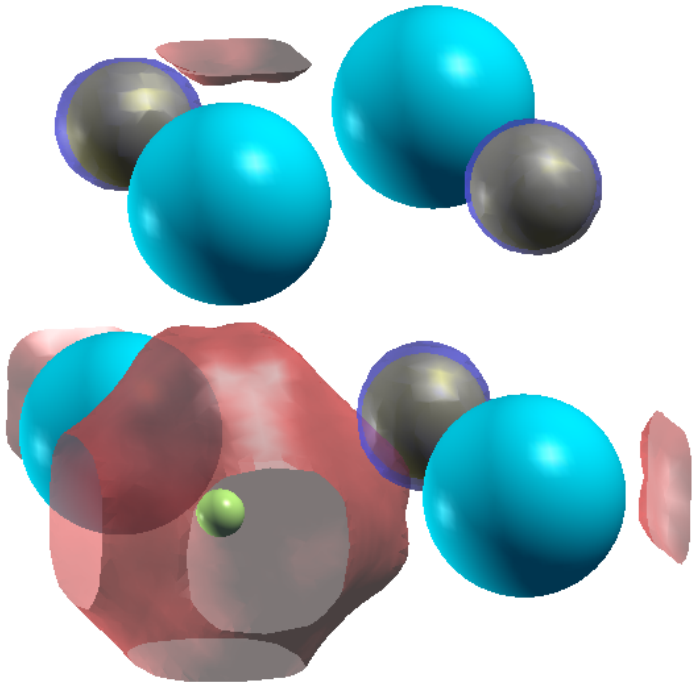


図 3: Cl サイト空孔を持った NaCl の最高占有状態の等値面 (Γ 点)。この等値面は Cl 空孔をもつ NaCl の F 中心を示している。等値面は XCrySDen を使用して、0.042 の等値 (isovalue) で描画 [61]。系の電荷を-1 として、キーワード「scf.system.charge」を使用して計算。水色および銀色はそれぞれナトリウムおよび塩素原子を表し、黄緑色の小丸は空原子の位置を示す。

OpenMX の計算に慣れるために有用だと思われます。

データベース Ver. 2013 の精度はデルタ因子により検証されています [27]。71 元素の平均デルタ因子は 1.6 meV/atom、標準偏差は 1.4 meV/atom となっており、これはデータベース Ver. 2013 の高い精度を示しています。したがって一般のユーザーはこれらの最適化された基底関数を使用することが推奨されます。デルタ因子の計算に関しては、「”格子定数 vs. エネルギー”」の章も参照して下さい。

9.4 ユーザーによる PAO の最適化

変分最適化法を用いれば、ユーザー自身で変分原理に基づき動径関数の形状を最適化することができます。詳細は「軌道の最適化」の章を参照して下さい。

9.5 空原子の配置

プリミティブ関数および最適化 PAO 関数は通常、原子に割り当てられます。さらに、**空原子**を使って空孔領域に基底関数を割り当てることも可能です。データベースのウェブサイト (http://www.openmx-square.org/) には空原子「E」の項目があります。空原子「E」の擬ポテンシャルはフラットなゼロポテンシャルですが、この空原子「E」の擬ポテンシャルを使用することで、原子位置とは独立に、空間上

の任意の場所に基底関数を配置することができます。それを行うために、空原子を以下のように定義します。

```
<Definition.of.Atomic.Species
  H    H5.0-s2p1      H_PBE13
  C    C5.0-s2p1      C_PBE13
  EH   H5.0-s2p1      E
  EC   C5.0-s2p1      E
Definition.of.Atomic.Species>
```

この例では、2 種の空原子が「EH」および「EC」として定義されており、それぞれは「H5.0-s2p1」および「C5.0-s2p1」で指定される基底関数が割て当てられています。したがって空原子に対して、任意の基底関数が使用可能であり、また空原子の定義も空原子に対する擬ポテンシャルとして「E.vps」を使用するだけで、簡単に行うことができます。そして「EH」および「EC」はキーワード「Atoms.SpeciesAndCoordinates」によって、任意の場所に配置することができ、そこでは空原子に対する擬ポテンシャルはゼロであり、また空原子から系に供給される電子数もゼロとなります。空原子の配置により、counterpoise 法 (CP) 法 [33, 34] を使って基底関数重なり誤差を推測できるだけでなく、線形結合擬原子軌道 (LCPAO) 法の範囲内で金属表面上の空孔状態および自由電子的状態を取り扱うこともできます。例として、Cl 空孔をもつ NaCl の F 中心の計算を図 3 に示します。Γ 点での最高占有状態がこの F 中心の状態であることがわかります。この計算は「work」ディレクトリ内の「NaCl _FC.dat」を使用して再現できます。

9.6 PAO および VPS ファイルを保存するディレクトリの指定

PAO および VPS ディレクトリへのパスは、以下のキーワードを使用して入力ファイル内で指定することができます。

```
DATA.PATH    ../DFT_DATA13    # default=../DFT_DATA13
```

絶対および相対パスの指定が可能です。基底関数と擬ポテンシャルのデータベースは複数のバージョンが公開されておりますが、いくつかの元素に対しては擬ポテンシャル中のセミコアの状態数がバージョン間で同一ではありません。したがって、これらのデータを使用する際には、同一バージョンの PAO および VPS を使用することが推奨されます。こうした理由から、PAO および VPS ファイルの各バージョンは異なったディレクトリに保存するのが良いでしょう。その際に、このキーワードが有用になります。

10　擬ポテンシャル

OpenMX では、原子核のクーロンポテンシャルは、Morrison、Bylander、Kleinman により提案されたノルム保存擬ポテンシャル [23] で置き換えられます。これは Vanderbilt によるウルトラソフト擬ポテンシャル [24] のノルム保存型となっています。擬ポテンシャルは、ADPACK を使用して生成できますが、ユーザーの利便性のために、擬ポテンシャルのデータベース Ver. 2013 が OpenMX の web サイトで公開されています (ADPACK は原子の密度関数計算を行うためプログラムパッケージであり、OpenMX の web サイトで入手可能です)。データベースに保存されている擬ポテンシャルを使用する場合には、それらをディレクトリ「openmx3.7/DFT_DATA13/VPS/」にコピーしてください。ただし、ほとんどのデータは既に OpenMX Ver. 3.7. の配布パッケージ内にコピー済みです。これらのデータは GNU-GPL に従って自由に利用することが可能ですが、利用に伴う損害・不利益等については一切責任を負いかねますので、あらかじめご了承下さい。擬ポテンシャルの指定は、基底関数の指定と同様にキーワード「Definition.of.Atomic.Species」で行います。例えば次の様にして指定できます。

```
<Definition.of.Atomic.Species
  H   H6.0-s2p1        H_CA13
  C   C6.0-s2p2        C_CA13
Definition.of.Atomic.Species>
```

擬ポテンシャルのファイルは第 3 列で指定でき、またそのファイルはディレクトリ「DFT_DATA13/VPS」に存在する必要があります。さらに原子座標の指定の際、各原子のアップおよびダウンスピン状態に対する電子数の指定を以下のように行うことが必要です。

```
<Atoms.SpeciesAndCoordinates
  1   C      0.000000    0.000000    0.000000     2.0  2.0
  2   H     -0.889981   -0.629312    0.000000     0.5  0.5
  3   H      0.000000    0.629312   -0.889981     0.5  0.5
  4   H      0.000000    0.629312    0.889981     0.5  0.5
  5   H      0.889981   -0.629312    0.000000     0.5  0.5
Atoms.SpeciesAndCoordinates>
```

ここで第 6、7 列は各原子のそれぞれアップおよびダウンスピン状態に対する初期電子数を与えます。原子に対するアップおよびダウン電子数の合計は、擬ポテンシャルの生成時に考慮される価電子の数と等価でなければなりません。それぞれの擬ポテンシャルに含まれている荷電子数は擬ポテンシャルファイル「*.vps」の中から見つけることができます。例えば、データベース Ver. 2013 内の炭素原子のファイル「C_PBE13.vps」の中に次の行があります。

```
 valence.electron            4.0
```

数字「4.0」は擬ポテンシャルの生成で考慮される電子数です。従って上記の例では、アップスピン電子数 (2.0) およびダウンスピン電子数 (2.0) の合計は「Atoms.SpeciesAndCoordinates」で指定する「C」に対して 4.0 となります。

ユーザー自身で ADPACK を使用して擬ポテンシャルを作成する場合には、以下の点に注意する必要があります。

- 非物理的なゴースト状態が生じていないかを確認すること。擬ポテンシャルの分離形式を使用しているため、ゴースト状態が生じる可能性があります。目的とする系の計算を行う前に、単純な系に対してベンチマーク計算を行い、擬ポテンシャルが適切かどうかを確認するべきです。
- 部分内殻補正のために滑らかな内殻電子密度を作成すること。そうでない場合には、数値積分の積分グリッドに対して高いカットオフエネルギーが必要となり、計算コストが増大します。

さらなる詳細はプログラムパッケージ「ADPACK」のマニュアルを参照して下さい。しかしながら、良い擬ポテンシャルを作成するには、最初に考える以上に多くの経験が必要になることを明記しておきます。

11　カットオフエネルギー：数値積分のための計算グリッドの設定

11.1　収束

計算精度と計算量は、数値積分およびポアソン方程式の解法で用いられるカットオフエネルギーに依存します [29]。カットオフエネルギーはキーワード「scf.energycutoff」により制御されます。図 4 にカットオフエネルギーに対するメタン分子の全エネルギーの収束の様子を示します。ここで入力ファイルは「入力ファイル」の章で使用された「Methane.dat」です。カットオフエネルギーは平面波法の場合のように基底系に対するものではなく、数値積分に対するものであることから、全エネルギーは平面波基底系の場合のようにカットオフエネルギーに応じて高エネルギー領域から収束していくわけではありません。ほとんどの場合、150 から 200 Ryd のカットオフエネルギーが最適な選択となります。しかしながら、300 Ryd 以上のカットオフエネルギーを必要とする微妙な問題があることにも注意すべきです。極小点付近で非常に平坦なポテンシャル形状の計算や異なるスピン秩序間の小さなエネルギー差の計算がそのような微妙な問題となる可能性があります。

表 1 に水分子の構造パラメータと双極子モーメントのカットオフエネルギーへの依存性を示します。入力ファイルは「work」ディレクトリ内の「H2O.dat」です。およそ 90 Ryd で収束した結果が得られています。必要なカットオフエネルギー値は元素に依存しますが、多くの場合に 200 Ryd 程度で収束解が得られるでしょう。しかしながら御自身の計算対象の系の物理的特性が、どの様にカットオフエネルギーに依存しているのかチェックされることを推奨します。もう一方のカットオフエネルギー「1DFFT.EnergyCutoff」に対しては、通常、デフォルト値として 3600 (Ryd) を使用しており、この値で十分に収束しています。またその計算量も他の部分の計算と比較し、小さなものです。

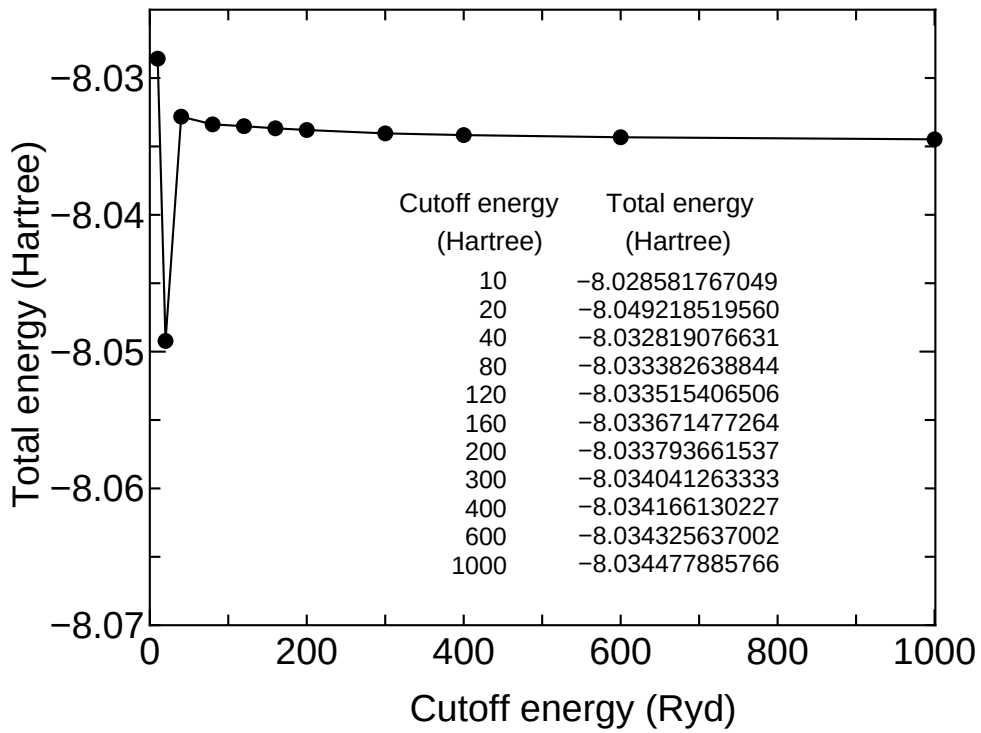


図 4: カットオフエネルギーに対するメタン分子の全エネルギーの収束性。

表 1: カットオフエネルギーに対する水分子の構造パラメータと双極子モーメントの収束性。入力ファイルは「work」ディレクトリ内の「H2O.dat」。

Ecut(Ryd)	r(H-O) (Å)	∠ (H-O-H) (deg)	Dipole moment (Debye)
60	0.970	103.4	1.838
90	0.971	103.7	1.829
120	0.971	103.7	1.832
150	0.971	103.6	1.829
180	0.971	103.6	1.833
Exp.	0.957	104.5	1.85

11.2 バルクのエネルギー曲線計算に関するヒント

格子定数の関数としてバルク系のエネルギー曲線を計算する際には、エネルギー曲線に不連続性が現れる場合があります。一般に、この不連続性は格子定数の変化に伴い積分に用いられている実空間グリッドのグリッド数が変化することによるものです。例として、図 5 に bcc-Fe のエネルギー曲線を示しますが、カットオフエネルギーが固定であるとき (200 および 290 (Ryd) の場合)、エネルギー曲線が不連続 (それほど顕著ではありませんが) になっていることが分かります。不連続なエネルギーの飛びは計算グリッド数が変化する格子定数において発生しています。エネルギー曲線上の不連続性を避けるために、キーワード「scf.Ngrid」を使うことができます。

```
scf.Ngrid     32 32 32    # n1, n2, and n3 for a-, b-, and c-axes
```

このキーワードによって実空間グリッド数が明示的に指定された場合には、キーワード「scf.energycutoff」で指定したカットオフを参照することなしに、ここで与えたグリッド数で各単位胞ベクトルが離散化されます。図 5 から分かるように、32×32×32 に固定されたグリッドによる計算では滑らかな曲線が得られています。

11.3 構造格子の相対位置の固定

OpenMX Ver. 3.7 では、キーワード「scf.energycutoff」もしくは「scf.Ngrid」で生成された実空間の積分グリッドは、一部のハミルトニアン行列要素の算出とポアソン方程式の解法に用いられています。ここで一部のハミルトニアン行列要素とは、差電子密度から生じる Hartree ポテンシャルと交換相関ポテンシャルに対する行列要素のことです。したがって全エネルギーは原子座標とグリッド間の相対位置に依存します。原子座標の関数として相互作用エネルギーやエネルギー曲線を計算する場合には、それらの一連の計算に必要な全ての計算において、この相対位置を保持することが、誤差を軽減するために重要です。相対位置を保持するためには、グリッド生成のための原点の x、y、z 成分を全ての計算において共通にする必要があります。標準出力には、次の様なグリッド生成時の原点となる x、y、z 成分が「Grid_Origin」として出力されています。

```
Grid_Origin  xxx  yyy  zzz
```

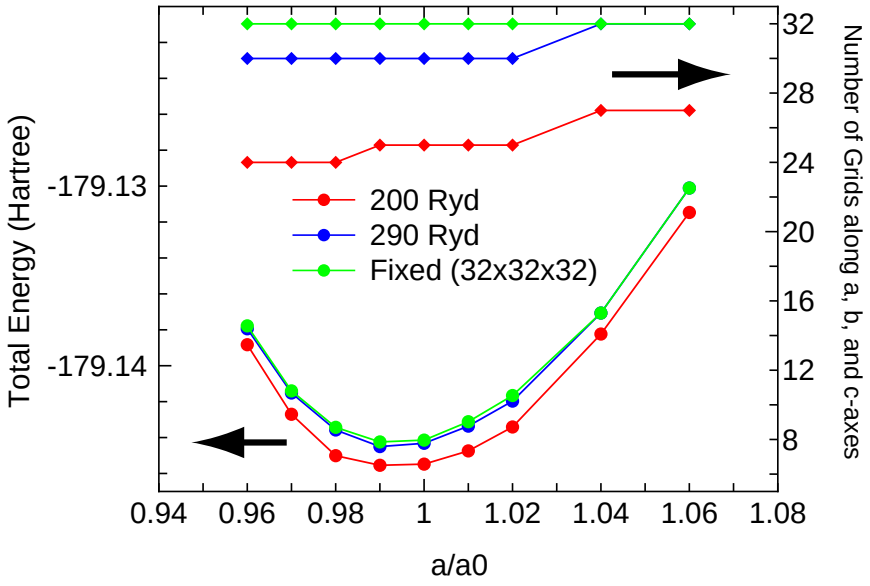


図 5: 格子定数の関数としての bcc 鉄の全エネルギー（実験による平衡格子定数 a_0 は 2.87Å）。2 原子を含む立方単位胞により計算。入力ファイルは「work」ディレクトリ内の「Febcc2.dat」。

一連の計算中のある一つの計算からこの値「xxx yyy zzz」を取得した後には、その他の全ての計算中で同一の値を使用しなければなりません。「Grid_Origin」を明示的に指定するためには、次のキーワード「scf.fixed.grid」によって「xxx yyy zzz」を指定して下さい。

```
scf.fixed.grid  xxx  yyy  zzz
```

この手続きにより、原子座標とグリッドとの相対位置をほぼ保持することが可能となり、積分グリッドの使用に伴う数値誤差が軽減されます。

さらに、前節「バルクのエネルギー曲線計算に関するヒント」で議論したように、一連の計算において格子定数が変化する場合には、キーワード「scf.Ngrid」によりグリッド数を明示的に固定して下さい。

12 SCF 収束

12.1 概要

OpenMX Ver. 3.7 では、以下の 5 種類の電子密度混合法がキーワード「scf.Mixing.Type」により利用可能です。

- 単純混合 (Simple)

  関連キーワード: scf.Init.Mixing.Weight, scf.Min.Mixing.Weight, scf.Max.Mixing.Weight

- 反復部分空間における最小残差法 (RMM-DIIS) [40]

  関連キーワード: scf.Init.Mixing.Weight, scf.Min.Mixing.Weight, scf.Max.Mixing.Weight, scf.Mixing.History, scf.Mixing.StartPulay

- Guaranteed reduction Pulay 法 (GR-Pulay) [39]

  関連キーワード: scf.Init.Mixing.Weight, scf.Min.Mixing.Weight, scf.Max.Mixing.Weight, scf.Mixing.History, scf.Mixing.StartPulay

- Kerker 混合 (Kerker) [41]

  関連キーワード: scf.Init.Mixing.Weight, scf.Min.Mixing.Weight, scf.Max.Mixing.Weight, scf.Kerker.factor

- Kerker 計量による RMM-DIIS (RMM-DIISK) [40]

  関連キーワード: scf.Init.Mixing.Weight, scf.Min.Mixing.Weight, scf.Max.Mixing.Weight, scf.Mixing.History, scf.Mixing.StartPulay, scf.Mixing.EveryPulay, scf.Kerker.factor

最初の 3 種類の方法においては、密度行列を混合することで入力密度行列が生成されます。密度行列は実空間での量とみなすことができ、（スピン）電子密度に容易に変換することが可能です。一方、残りの 2 種類の方法では、フーリエ空間で電子密度混合が行われます。「RMM-DIIS」、 「GR-Pulay」、「RMM-DIISK」による混合法は全て Pulay 型混合法です。一般的に、大きなギャップを持つ系ではどの混合法を使用しても SCF 収束を達成することが可能です。しかし、小さなギャップを持つ系や金属系では、SCF 計算における電子密度の振動の問題 (charge sloshing) がしばしば深刻となることから、SCF 収束を達成することは容易ではありません。そのような困難な系に対応するために、「Kerker」および「RMM-DIISK」による 2 種の混合法が利用可能です。この 2 種の混合法は金属系の SCF 収束を達成するための有効な方法です。「Kerker」または「RMM-DIISK」を使用する際には、SCF 計算の収束を加速するために以下の処方が有用です。

- 「scf.Mixing.History」を増やす。30 から 50 の比較的大きな値によって収束しやすくなります。さらに「scf.Mixing.EveryPulay」は 1 に設定されなければなりません。

- 「scf.Mixing.StartPulay」に、いくらか大きめの値を使用する。Pulay 型の混合を開始する前に、ある程度の収束を達成しなければなりません。「scf.Mixing.StartPulay」の適切な値はおそらく 10 から 30 でしょう。

- 金属系の場合には「scf.ElectronicTemperature」にいくらか大きめの値を使用する。
  「scf.ElectronicTemperature」が小さい場合には、しばしば数値的不安定性が発生します。

SCF 計算における電子密度の振動は一般に電子密度の長波長成分から生じています。波数 $\mathbf{q}$ の関数である Kerker 重み $w_{\mathbf{q}}$ を次式の内積計算に導入することで、この長波長成分からの振動を抑制することができます。ここで振動の抑制度合は Kerker 因子 α を調整することで制御され、その値はキーワード「scf.Kerker.factor」で与えることができます。

$$\langle A|B\rangle = \sum_{\mathbf{q}} \frac{A^*_{\mathbf{q}} B_{\mathbf{q}}}{w_{\mathbf{q}}}$$

$$w_{\mathbf{q}} = \frac{|\mathbf{q}|^2}{|\mathbf{q}|^2 + q_0^2}$$

$$q_0 = \alpha |\mathbf{q}_{\min}|$$

ここで、$\mathbf{q}_{\min}$ は 0 ベクトルを除く最小の大きさを持つ $\mathbf{q}$ ベクトルです。大きな α は電子密度の振動を大幅に抑制しますが、収束は遅くなる可能性があります。最適値は系に依存しますので、計算対象の系に適した値を、ユーザーが調整することが必要です。

さらに、「RMM-DIISK」の振る舞いは次のキーワードにより調整することができます。

```
scf.Mixing.EveryPulay    5   # default = 1
```

Pulay 型混合法の残差ベクトルは、混合ステップが累積するにつれてある特定の部分空間を張るだけとなり、このため、その部分空間に直交した線型独立な新しいベクトル成分が導入されないために収束が困難になってきます。この線形従属問題を回避する一つの方法は、Kerker 混合の合間に時折 Pulay 型混合を行うことです。この方法ではキーワード「scf.Mixing.EveryPulay」を使用して頻度を指定することができます。例えば「scf.Mixing.EveryPulay=5」の場合、5 回の SCF 反復ごとに Pulay 混合が、他のステップでは Kerker 型混合が行われます。「scf.Mixing.EveryPulay=1」は従来の Pulay 型混合に対応します。キーワード「scf.Mixing.EveryPulay」は「RMM-DIISK」に対してのみ使用でき、デフォルト値は「1」であることに注意して下さい。

SCF 収束性を向上させるための上記の処方箋は多くの場合に有効です。しかし、収束を加速させるために最も推奨されるのは以下の方法です。

- 「scf.Mixing.History」を増やす。30 から 50 の比較的大きな値によって収束しやすくなります。
  さらに「scf.Mixing.EveryPulay」は 1 に設定しなければなりません。

RMM-DIIS や RMM-DIISK などの Pulay 型混合は準ニュートン法に基づいていることから、収束速度はいかに適切な近似ヘッセ (Hessian) 行列を見つけられるかに依存します。「scf.Mixing.History」を大きくするに従い、計算される近似ヘッセ行列の精度が向上する可能性があります。

図 6 は (a) シアル酸分子、(b) Pt_{13} クラスタ、 (c) Pt_{63} クラスタに対する 5 種類の混合法の SCF 収束の比較を示しています。密度行列もしくは電子密度の残差ノルムは「*.out」ファイル内で NormRD として記録されています。またこの計算で用いた入力ファイルは「work」ディレクトリ内の「SialicAcid.dat」、

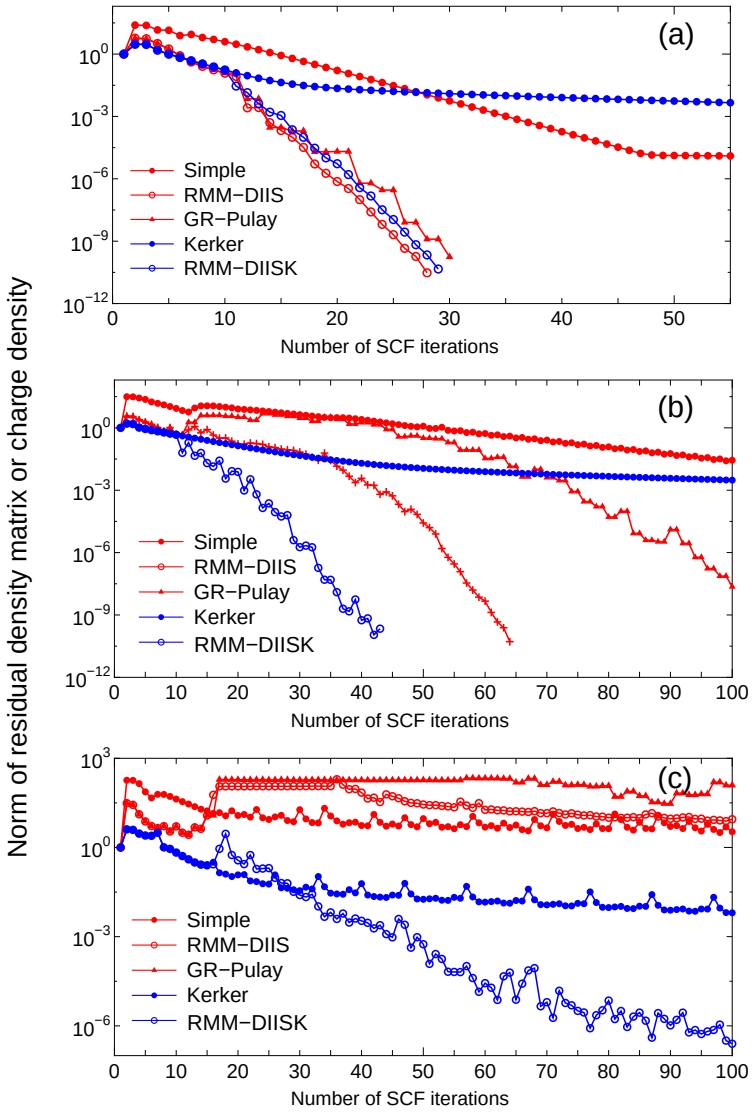


図 6: 5 種類の電子密度混合法における密度行列もしくは電子密度の残差ノルムの SCF 収束の過程。(a) シアル酸分子、(b) Pt_{13} クラスタ、 (c) Pt_{63} クラスタ。入力ファイルは「work」ディレクトリ内の「SialicAcid.dat」, 「Pt13.dat」、「Pt63.dat」。

「Pt13.dat」、「Pt63.dat」です。図 6 に示された全ての系に対して「RMM-DIISK」が頑健に動作していることがわかります。SCF 収束が極めて困難な場合では、大きな「scf.Kerker.factor」および小さな「scf.Max.Mixing.Weight」を用いた「Kerker」の使用が必要となりますが、多くの場合で、「RMM-DIISK」が最良の選択となります。

12.2 Kerker 因子の自動決定

入力ファイルでキーワード「scf.Kerker.factor」が与えられていない場合、OpenMX Ver. 3.7 は次式により Kerker 因子 α の適切な値を自動的に推定します。

$$\alpha = \frac{0.5}{|\mathbf{b}_{\rm min}|^2}\left(4\frac{Dq}{Aq}+1.0\right),$$

このとき、

$$Aq = \frac{1}{3}\left(|\mathbf{b}_1|^2+|\mathbf{b}_2|^2+|\mathbf{b}_3|^2\right),$$

$$Dq = \frac{1}{3}\sum_{i<j}\left||\mathbf{b}_i|^2-|\mathbf{b}_j|^2\right|,$$

となります。ここで、$\mathbf{b}_i (i=1,2,3)$ は逆格子ベクトルであり、$\mathbf{b}_{\rm min}$ は $\{\mathbf{b}\}$ 内での最小ベクトルです。この式では系のサイズと異方性に対する α の依存性が考慮されています。一連の数値計算から、推測した値はほとんどの場合に、良好に機能していることが分かっています。

12.3 SCF 収束パラメータの on-the-fly での調整

SCF 計算を行っている途中で SCF 収束に関する以下のパラメータを変更することが可能です。

```
scf.maxIter
scf.Min.Mixing.Weight
scf.Max.Mixing.Weight
scf.Kerker.factor
scf.Mixing.StartPulay
```

例えば、入力ファイルで次の 2 種類のキーワードを以下のように指定した場合には、

```
System.CurrrentDirectory          ./    # default=./
System.Name                       c60
```

ディレクトリ「./」内に「c60_SCF_keywords」という名前のファイルを作成して下さい。このファイル中に例えば、以下のように記述します。

```
scf.maxIter                100
scf.Min.Mixing.Weight      0.01
scf.Max.Mixing.Weight      0.10
scf.Kerker.factor          10.0
scf.Mixing.StartPulay      30
```

OpenMX は SCF ステップ毎にファイル「c60_SCF_keywords」をチェックします。ファイルの読み込みに成功した場合には、標準出力に次のメッセージが表示されます。

```
The keywords for SCF iteration are renewed by ./c60_SCF_keywords.
```

また、キーワード「scf.maxIter」に対して負の値が与えられた場合には、OpenMX は終了します。大規模計算を行う場合、on-the-fly での SCF 収束パラメータの調整は有用でしょう。

13　再スタート

13.1　概要

状態密度、バンド分散、分子軌道などの計算を行う場合には予め自己無撞着計算を行っておき、二回目以降の計算では自己無撞着計算をスキップすれば、計算時間を節約することができます。この計算の再スタートを行うためのキーワードが「scf.restart」として用意されています。

```
scf.restart             on          # on|off,default=off
```

キーワード「scf.restart」を「on」にすると、最初の計算で作成された再スタート用のファイルが読み込まれ、二番目の計算での初期のハミルトニアンおよび (スピン) 電子密度として使用されます。ここで二番目の計算の「System.Name」は一番目の計算と同一でなければなりません。再スタートファイルは「work」ディレクトリの下の「*_rst」ディレクトリ内に保存されています。ここで*は「System.Name」を意味します。「*_rst」内の再スタートファイルは密度行列混合法およびフーリエ空間混合法の両方に関係する全ての情報を含んでいます。従って、二番目の計算において別の混合法を使うこともできます。再スタートの例として、図 7 に SCF 計算の過程を示します。この計算は「work」ディレクトリ内にある入力ファイル「C60.dat」を用いて行ったものです。図 7 より、二番目の計算では再スタートファイルの使用により SCF 計算がすぐに終了していることが分かります。

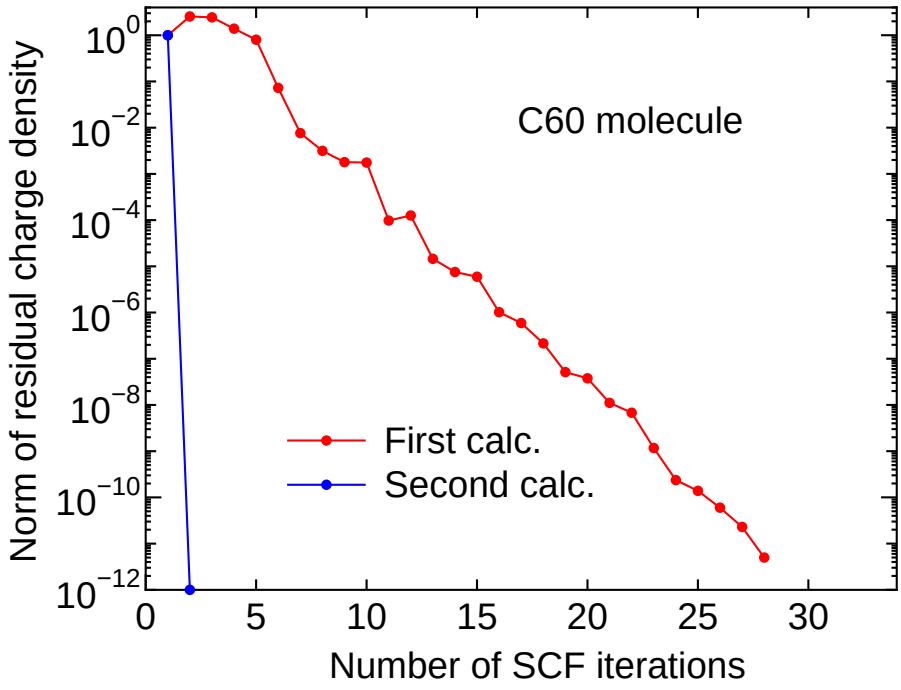


図 7: C_{60} 分子の SCF 収束。二番目の計算では、最初の計算により生成された再スタートファイルを使用。入力ファイルは「work」ディレクトリ内にある「C60.dat」。

13.2 MD および構造最適化中の外挿法

構造最適化および分子動力学 (MD) シミュレーションでは、SCF の収束を加速するために、過去のステップで生成された再スタートファイルが自動的に利用されます [42, 43]。この方法は電子密度の外挿法に基づいており、外挿に用いる過去のステップ数をキーワード「scf.ExtCharge.History」により設定できます。

```
scf.ExtCharge.History      2        # default=2
```

一連のベンチマーク計算から、「scf.ExtCharge.History」を 2 に設定すると収束が加速し、それ以上の大きな値を用いると数値的に不安定となることが分かっています。したがって、デフォルト値の 2 を使用することを推奨します。

13.3 再スタート計算用の入力ファイル

構造最適化および分子動力学 (MD) シミュレーションの各ステップにおいて、再スタート計算用の入力ファイル「*.dat#」が出力されます。「*.dat#」には最終ステップでの原子座標が挿入されており、また「構造格子の相対位置の固定」の章で説明された「Grid_Origin」の設定も行われています。この入力ファイル「*.dat#」を用いて構造最適化および分子動力学 (MD) シミュレーションを前回の最終ステップから継続することが可能です。

14　構造最適化

14.1　最急降下法

構造最適化の例を本章で説明します。初期構造として、「入力ファイル」の章で扱ったメタン分子を取り上げますが、メタン分子の炭素原子の x 座標が以下のように 0.3 Å に変更されています。

```
<Atoms.SpeciesAndCoordinates
  1   C      0.300000     0.000000     0.000000      2.0   2.0
  2   H     -0.889981    -0.629312     0.000000      0.5   0.5
  3   H      0.000000     0.629312    -0.889981      0.5   0.5
  4   H      0.000000     0.629312     0.889981      0.5   0.5
  5   H      0.889981    -0.629312     0.000000      0.5   0.5
Atoms.SpeciesAndCoordinates>
```

次に、キーワード「MD.type」を「Opt」に、キーワード「MD.maxIter」を 200 に設定します。「Opt」は可変の前因子を持った最急降下法です。メタン分子の構造最適化の収束履歴を図 8(a) に示します。図では原子にかかる力の最大絶対値を構造最適化のステップ数の関数として示しています。この最急降下法では収束の状況に応じて前因子を変化させるため、前因子が大きくなりすぎて構造が停留点を越えていることが見て取れます。その後で完全な収束に至っています。この計算は「work」ディレクトリ内の「Methane2.dat」を使用することにより、再現することができます。またメタン分子の場合と同様、図 8(b) に示すようにダイアモンド構造のケイ素に対しても類似した挙動を見ることができます。

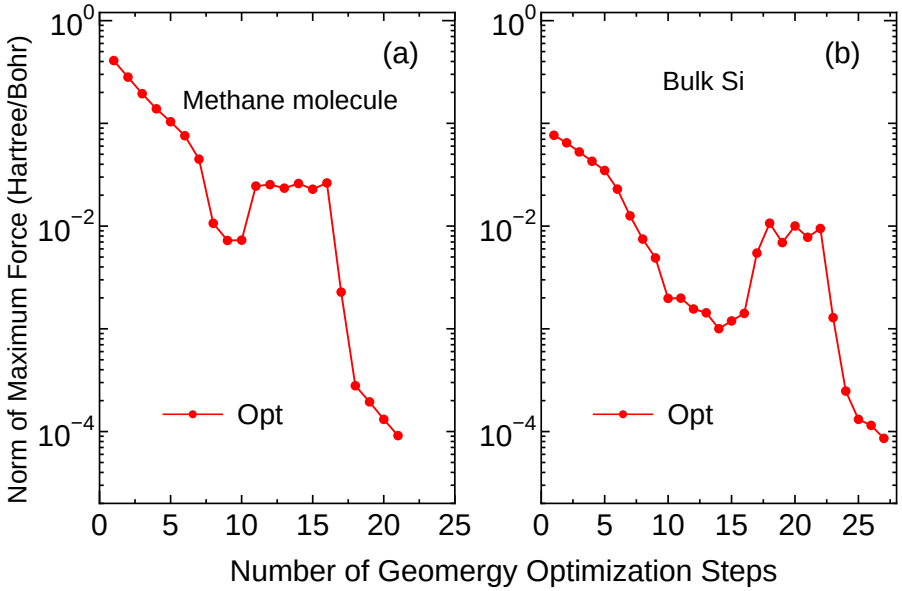


図 8: (a) メタン分子 と (b) ダイアモンド構造のケイ素の原子の構造最適化における原子にかかる力の絶対最大値。初期構造は平衡構造から変形されたものを使用。入力ファイルはそれぞれ、「work」ディレクトリ内の「Methane2.dat」および「Si8.dat」。

14.2 EF、BFGS、RF、DIIS 最適化

「Opt」は頑健な方法ですが、一般的に収束速度は遅いと言えます。より高速な方法として、準ニュートン法に基づいた構造最適化手法が利用可能です。デカルト座標内で実行される固有ベクトル追跡 (EF) 法 [45]、Broyden-Fletcher-Goldfarb-Shanno (BFGS) 法 [47]、有理関数 (RF) 法 [46]、そして反復部分空間 (DIIS) 法 [44] がサポートされています。EF および RF 法では、近似ヘシアンは BFGS 法により更新されます。キーワード「MD.Type」により、「Opt」、「EF」、「BFGS」、「RF」、「DIIS」の一つを選択して下さい。関連キーワードは次の通りになります。

```
MD.Type                    EF          # Opt|DIIS|BFGS|RF|EF
MD.Opt.DIIS.History         3          # default=3
MD.Opt.StartDIIS            5          # default=5
MD.Opt.EveryDIIS          200          # default=200
MD.maxIter                100          # default=1
MD.Opt.criterion        1.0e-4         # default=0.0003 (Hartree/Bohr)
```

特に、これらの準ニュートン法に基づいた構造最適化法は次の 2 種のキーワードにより制御することができます。

```
MD.Opt.DIIS.History     3       # default=3
MD.Opt.StartDIIS        5       # default=5
```

キーワード「MD.Opt.DIIS.History」は近似ヘシアンを更新するための過去の履歴ステップ数を指定します。デフォルト値は 3 です。また、「EF」、「BFGS」、「RF」、「DIIS」を開始する構造最適化ステップをキーワード「MD.Opt.StartDIIS」により指定します。これらの方法を開始する前の構造最適化ステップは、最急降下法により実行されます。デフォルト値は 5 です。

最適化の初期ステップは初期構造における最大の力を参照することで自動的に調整されます。図 9 は、分子やバルクにおいて最大力が 0.0003 Hartree/Bohr 以下になるまでの構造最適化ステップ数を示しています。ここに示されるように、EF および BFGS 法も同様な性能を示しているものの、ほとんどの場合に RF 法が最もロバストで効率的な方法です。これらの計算に使用された入力ファイルと出力ファイルは「work/geoopt_example/」ディレクトリ内にあります。

また、これらの準ニュートン法により、構造が最小停留点よりもむしろ鞍点に収束される可能性もあることに注意すべきです。これは、準ニュートン法の使用が開始された時の構造が変曲点に達しない場合に起り得ます。そのような場合には、準ニュートン法に移る前に、構造を最急降下法により十分に最適化するべきです。対処方法は、「MD.Opt.StartDIIS」に対して大きな値を使用するだけです。十分な収束に対して多数の反復ステップが要求される系では、ファイル「*.dat#」を使用して計算を再スタートすることも必要かも知れません。ここで「*」は入力ファイルで指定した「System.Name」です。

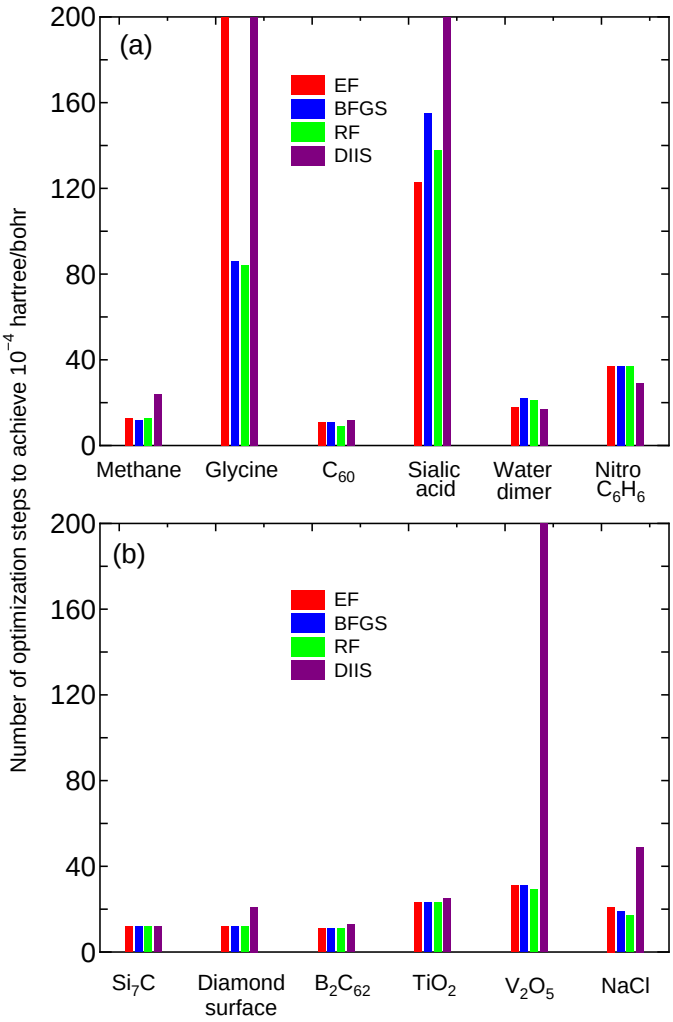


図 9: (a) 分子系および (b) バルク系に対する構造最適化の反復回数。収束判定条件は原子に働く力の絶対最大値が 3×10^{-4} Hartree/Bohr 以下。

14.3 制約条件付の構造最適化

構造最適化の際に任意の原子座標を初期座標に固定することができます。この制約条件はキーワード「MD.Fixed.XYZ」を用いて各原子の x、y、z 座標に対して、以下のように設定します。

```
<MD.Fixed.XYZ
   1  1 1 1
   2  1 0 0
 MD.Fixed.XYZ>
```

上記の例は 2 つの原子から構成される系の場合です。系に N 個の原子がある場合、N 個分の指定が必要です。第 1 列は「Atoms.SpeciesAndCoordinates」で個々の原子を特定するのに使用した連番です。第 1 列は「Atoms.SpeciesAndCoordinates」で個々の原子を特定するのに使用した連番です。第 2〜4 列は、

それぞれ x、y、z 座標のフラグです。フラグが「1」だと座標は固定、「0」だと緩和されます。上記の例では、原子 1 の x、y、z 座標は固定、原子 2 は x 座標のみ固定されています。デフォルトでは全ての座標が緩和される指定になっています。このキーワードによる原子座標の固定は、すべての構造最適化および分子動力学の方法において有効です。制限条件付の構造最適化は大規模な系の局所構造の精密化にとって有用でしょう。

14.4 構造最適化の再スタート

最初の構造最適化の試行により、収束条件に達しなかった場合には、入力ファイル「*.dat#」を使って構造最適化を再スタートすることができます。入力ファイル「*.dat#」は構造最適化の各ステップで生成され、その最終ステップでの幾何構造が挿入されています。構造最適化を再スタートさせるには最終ステップでの幾何構造を用いるだけでなく、最初の試行で計算された近似ヘシアンを用いることも必要です。OpenMX Ver. 3.7 では、近似ヘシアンも構造最適化ステップ毎に保存され、「*.dat#」により再スタートを行うときに自動的に再利用されます。そのため、「*.dat#」を用いて複数の計算によって断続的に構造最適化が行われたとしても、構造最適化ステップのための反復回数は単一計算の場合と同じになります。実行時間に制限がある共用計算システムを用いて、大規模系を最適化する場合にはこの機能が有用となるでしょう。

15　分子動力学

OpenMX Ver 3.7 は次の 5 種類の分子動力学シミュレーションをサポートしています。

- 定エネルギーの分子動力学 （NVE）
- 速度スケーリングによる定温 (NVT) 分子動力学 （NVT_VS）
- 各原子に対して独立に速度スケーリングを考慮した定温 (NVT) 分子動力学 （NVT_VS2）
- 能勢-Hoover 法による定温 (NVT) 分子動力学 （NVT_NH）
- 多重熱浴分子動力学 （NVT_VS4）

各分子動力学シミュレーションの詳細を以下に説明します。

15.1　定エネルギーの分子動力学

キーワード「MD.Type」を「NVE」と指定することで、定エネルギーの分子動力学シミュレーションが実行可能です。

```
MD.Type           NVE       # NOMD|Opt|NVE|NVT_VS|NVT_VS2|NVT_NH
```

MD の各ステップで計算される数値は、出力ファイル「*.ene」（*は「System.Name」を意味します）に記録されます。詳細は、「source」ディレクトリ内のファイル「iterout.c」に書かれていますが、いくつかの数値に関して、ここにも示しておきます。

```
 1:    MD step
 2:    MD time (fs)
14:    kinetic energy of nuclear motion, Ukc (Hartree)
15:    DFT total energy, Utot (Hartree)
16:    Utot + Ukc (Hartree)
17:    Fermi energy (Hartree)
```

出力ファイル「*.ene」中で、最初の列は MD Step（MD ステップ）、2 番目の列は MD time（MD 時間）等に対応しています。

15.2　速度スケーリングによる NVT 分子動力学

キーワード「MD.Type」を「NVT_VS」と指定することで、 速度スケーリング法 [17] による NVT アンサンブル分子動力学シミュレーションが実行可能です。

```
MD.Type           NVT_VS    # NOMD|Opt|NVE|NVT_VS|NVT_VS2|NVT_NH
```

この NVT 分子動力学では原子運動の温度は、以下のような書式で制御することができます。

```
<MD.TempControl
  3
  100    2  1000.0  0.0
  400   10   700.0  0.4
  700   40   500.0  0.7
MD.TempControl>
```

この書式の最初の行は「<MD.TempControl」であり、最後の行は「MD.TempControl>」とする必要があります。最初の「3」は温度制御のために必要な行の数です。この場合には、引き続く 3 行で温度の指定を行っており、最初の列の数字は MD のステップ数を表し、2 列目の数字は速度スケーリングが行われる MD ステップの間隔を与えます。この計算例では、速度スケーリングは、MD ステップが 100 回までは 2 回毎に行われ、MD ステップの 100 回目から 400 回目までは 10 回ごとに行われ、さらに MD ステップ 400 回から 700 回では 40 回ごとに行われます。3 番目と 4 番目の列は、それぞれ設定温度 $T_{\rm give}$ (K) とその区間でのスケーリング・パラメータ α を表します。またその区間中における温度は線形補間によって与えられます。この例の速度スケーリングでは、速度は次式を用いてスケーリングしています。

$$s = \sqrt{\frac{T_{\rm given} + (T_{\rm calc} - T_{\rm given}) * \alpha}{T_{\rm calc}}}$$

$$\mathbf{v}_i' = \mathbf{v}_i \times s$$

ここで、$T_{\rm given}$ および $T_{\rm calc}$ はそれぞれ設定温度と原子の運動から計算された温度です。「NVT_VS」法では、全原子の速度を使って温度を計算します。一方、**局所**温度は「NVT_VS2」では、各原子の速度を用いて評価され、速度スケーリングはその局所温度に基づき行われます。「MD.TempControl」で指定された最後の MD ステップが終了すると、NVT アンサンブルは NVE アンサンブルに切り替わります。各 MD ステップで計算された数値は、出力ファイル「*.ene」(「*」は「System.Name」を意味します）に記録されます。詳細は、「source」ディレクトリ内のファイル「iterout.c」に書かれていますが、いくつかの数値に関して、ここにも示しておきます。

```
 1:    MD step
 2:    MD time (fs)
14:    kinetic energy of nuclear motion, Ukc (Hartree)
15:    DFT total energy, Utot (Hartree)
16:    Utot + Ukc (Hartree)
17:    Fermi energy (Hartree)
18:    Given temperature for nuclear motion (K)
19:    Calculated temperature for nuclear motion (K)
22:    Nose-Hoover Hamiltonian (Hartree)
```

出力ファイル「*.ene」中で、最初の列は MD Step（MD ステップ)、2 番目の列は MD time（MD 時間）等に対応しています。例として、図 10 (a) にグリシン分子の速度スケーリングによる MD 計算の結果を示します。分子の温度が設定温度の周辺で振動していることがわかります。分子動力学シミュレーションの様子を可視化するには、フリーソフト xmakemol [91] や XCrySDen [61] 等を用いて、出力ファイル「*.md」を簡単にアニメーション化することが可能です。

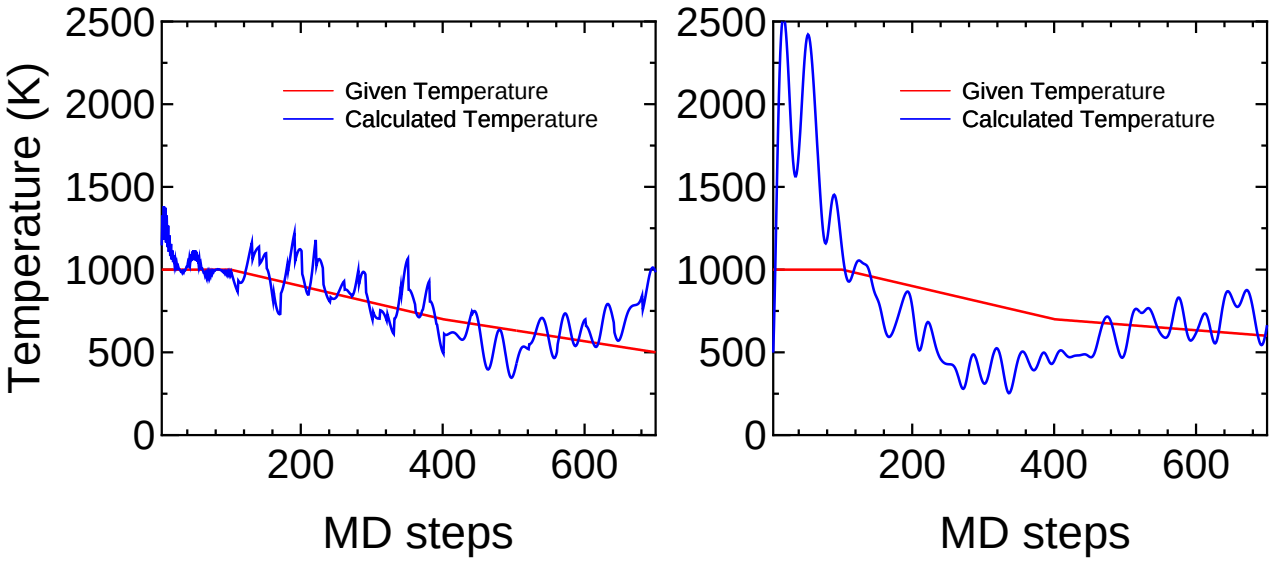


図 10: (a) 速度スケーリング NVT 分子動力学によるグリシン分子の設定温度と計算温度の時間変化。(b) Nose-Hoover NVT 分子動力学によるグリシン分子の設定温度と計算温度の時間変化。入力ファイルは、「work」ディレクトリ内のそれぞれ「Gly_VS.dat」および「Gly_NH.dat」。

15.3 Nose-Hoover 法による NVT 分子動力学

キーワード「MD.Type」を「NVT_NH」と指定することで、能勢-Hoover 法 [18] による NVT アンサンブル分子動力学シミュレーションが実行可能です。

```
MD.Type          NVT_NH      # NOMD|Opt|NVE|NVT_VS|NVT_NH
```

この NVT 分子動力学では原子運動の温度は、以下のような書式で制御することができます。

```
<MD.TempControl
  4
  1     1000.0
  100   1000.0
  400    700.0
  700    600.0
MD.TempControl>
```

この書式の最初の行は「<MD.TempControl」であり、最後の行は「MD.TempControl>」とする必要があります。最初の「4」は温度制御のために必要な行の数です。この場合には、引き続く 4 行で温度の指定を行っており、最初の列の数字は MD のステップ数を表し、2 列目は原子運動の設定温度を表します。MD ステップの間の温度は線形補間により与えられます。この場合でも、速度スケーリング MD で用いた同じキーワード「MD.TempControl」が使用されていますが、書式が異なることに注意して下さい。「MD.TempControl」の指定の他に、次のキーワードを用いて、熱浴の質量を指定して下さい。

```
NH.Mass.HeatBath          30.0       # default = 20.0
```

この設定では、炭素原子の主な同位体の質量を 12.0 とする統一原子質量単位を用いています。「速度スケーリングによる NVT 分子動力学」で説明したように、MD ステップの各段階において計算された数値は、出力ファイル名「*.ene」に保存されます。例として、Nose-Hoover 法によるグリシン分子の計算結果を図 10 (b) に示します。原子運動の温度が設定温度の周辺で振動していることがわかります。また、分子動力学の様子を可視化するには、NVT_VS の場合と同様に、フリーソフト xmakemol [91] や XCrySDen [61] 等を用いて、出力ファイル「*.md」を簡単にアニメーション化することが可能です。

15.4 多重熱浴分子動力学

OpenMX Ver. 3.7 では多重熱浴分子動力学計算（multi-heat bath molecular dynamics simulation）が実行可能です。グループ化した各原子の温度は速度スケーリング法 [17] による熱浴で制御します。この方法は次のキーワードで実行します。

```
MD.Type           NVT_VS4
```

グループ数は、

```
MD.num.AtomGroup    2
```

で指定し、グループは次のように定義されます。

```
<MD.AtomGroup
  1  1
  2  1
  3  1
  4  2
  5  2
MD.AtomGroup>
```

最初の行は「<MD.AtomGroup」、最後の行は「MD.AtomGroup>」でなければなりません。最初の列は原子を識別するための通し番号で、「Atoms.SpeciesAndCoordinates」で指定された番号に対応します。2 列目は各原子が属するグループを表す番号です。この番号は 1 から始まり、2、3、· · · という順序で指定しなければなりません。上の例は、系が 5 つの原子のみから構成され、2 つのグループに分けた場合です。Ver. 3.7 では、全グループの温度プロファイルは「速度スケーリングによる NVT 分子動力学」の節で述べたように、「MD.Temp.Control」というキーワードで制御します。将来的には、グループ毎に独立に温度が制御できるような機能も導入する予定です。

15.5 制約条件付き分子動力学

原子座標を初期位置に固定して分子動力学計算を行うための制約 MD シミュレーションが実行可能です。「制約条件付の構造最適化」の節で述べたのと同じ方法で設定します。指定方法はそちらの節を参照してください。

15.6 初速度

次のキーワードにより、分子動力学計算において、各原子の初速度を指定することが可能です。

```
<MD.Init.Velocity
 1    3000.000  0.0  0.0
 2   -3000.000  0.0  0.0
MD.Init.Velocity>
```

この例は、2 個の原子から構成される系の場合です。N 個の原子の系の場合は、N 行の指定行が必要となります。最初の列は、キーワード「Atoms.SpeciesAndCoordinates」の指定と同じ原子の通し番号です。2 列目、3 列目、4 列目の数値は、それぞれ、各原子の初期速度の x、y、z 成分です。速度の単位は m/s です。キーワード「MD.Init.Velocity」は「MD.Fixed.XYZ」と併用することが可能です。

15.7 ユーザーによる原子の質量の定義

OpenMX の分子動力学計算において、原子質量は「Set_Atom_Weight() of SetPara_DFT.c」で定義されています。しかし、キーワード「Definition.of.Atomic.Species」を用いて、簡単に原子質量を変更することが出来ます。その場合、次のように 4 列目の数値で原子質量を定義します。

```
<Definition.of.Atomic.Species
 H   H5.0-s1          H_PBE13      2.0
 C   C5.0-s1p1        C_PBE13     12.0
Definition.of.Atomic.Species>
```

4 列目の数値が明示的に与えられない際には、デフォルトの原子質量が使用されます。この方法は分子動力学計算で原子質量の影響を調べる際に有用でしょう。また、特に水素原子に対して重水素の質量を割り当てることで、より時間ステップを大きくすることが出来るようになります。原子質量の定義には、炭素原子の主同位体を 12.0 とする統一原子質量単位を用いています。

16　可視化

電子密度、分子軌道、ポテンシャル等のデータは Gaussian cube 形式でファイルに出力されます。図 11 に XcrySDen [61] を用いて可視化した等値面マップの例を示します。これらのデータは、Gaussian cube 形式で出力されているため、Molekel [60] や XCrySDen [61] 等の多くのソフトウエアで簡単に可視化できます。

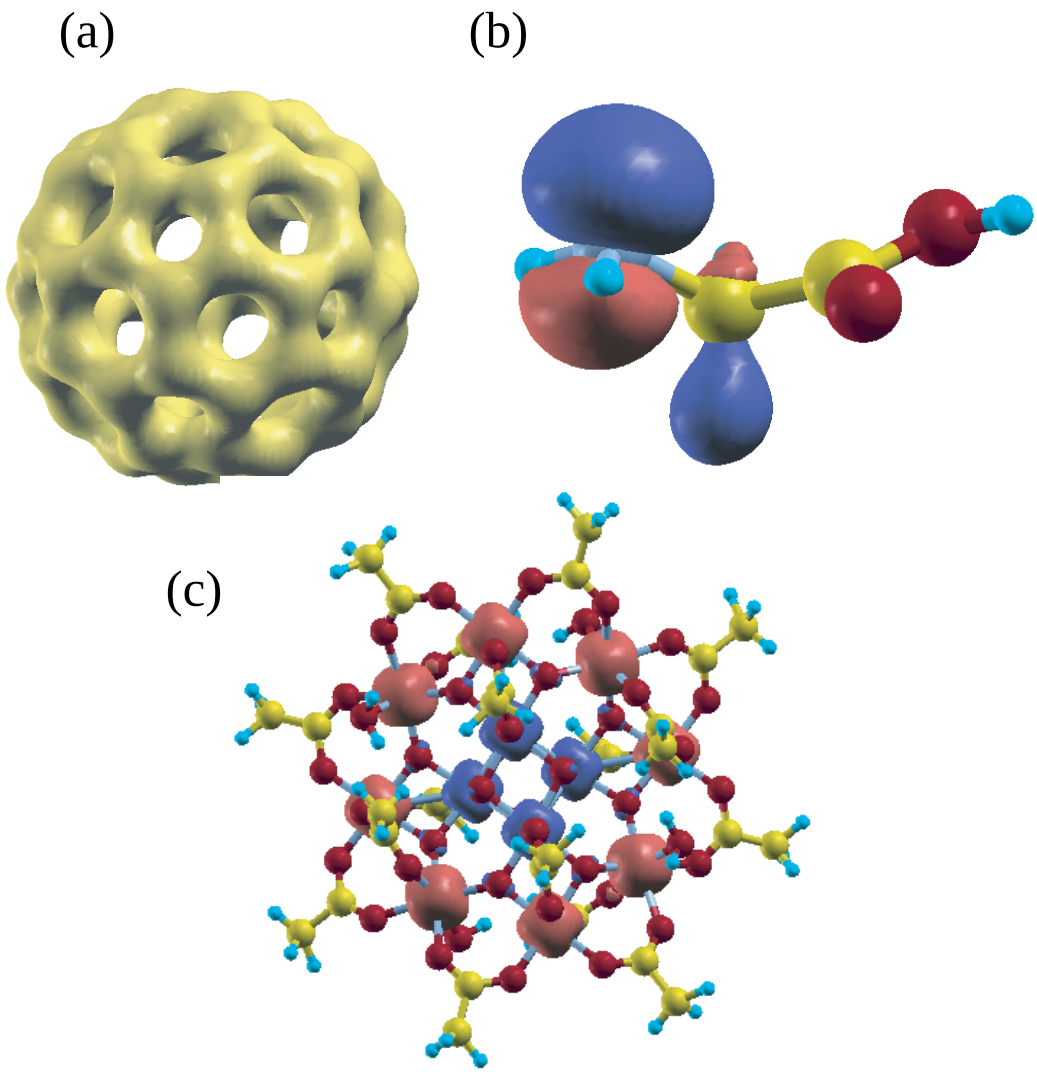


図 11: (a) C_{60} 分子の全電子密度の等値面 （0.13 の等値を使用）。(b) グリシン分子の最高被占軌道（HOMO）の等値面 （|0.06| の等値を使用）。(c) 分子磁石（$Mn_{12}O_{12}(CH_3COO)_{16}(H_2O)_4$ [62]）スピン電子密度の等値面（|0.02| の等値を使用）。

17　バンド分散

バンド分散は次の２つの段階を経由して、計算されます。

(1) SCF 計算

ダイヤモンド構造の炭素を例として、バンド分散の計算方法を説明します。「work」ディレクトリ内のファイル「Cdia.dat」には、原子の座標、単位胞ベクトルおよび「scf.Kgrid」が次のように指定されています。

```
Atoms.Number         2
Atoms.SpeciesAndCoordinates.Unit   Ang # Ang|AU
<Atoms.SpeciesAndCoordinates
 1   C  0.000  0.000  0.000   2.0 2.0
 2   C  0.890  0.890  0.890   2.0 2.0
Atoms.SpeciesAndCoordinates>
Atoms.UnitVectors.Unit             Ang # Ang|AU
<Atoms.UnitVectors
  1.7800  1.7800  0.0000
  1.7800  0.0000  1.7800
  0.0000  1.7800  1.7800
Atoms.UnitVectors>

scf.Kgrid                  7 7 7       # means n1 x n2 x n3
```

バンド分散の単位胞ベクトルとバンド分散を計算するための **k** 点の経路は次の書式で与えられます。

```
Band.dispersion             on        # on|off, default=off
<Band.KPath.UnitCell
 3.56  0.00  0.00
 0.00  3.56  0.00
 0.00  0.00  3.56
Band.KPath.UnitCell>
Band.Nkpath                  5
<Band.kpath
  15  0.0 0.0 0.0   1.0 0.0 0.0   g X
  15  1.0 0.0 0.0   1.0 0.5 0.0   X W
  15  1.0 0.5 0.0   0.5 0.5 0.5   W L
  15  0.5 0.5 0.5   0.0 0.0 0.0   L g
  15  0.0 0.0 0.0   1.0 0.0 0.0   g X
Band.kpath>
```

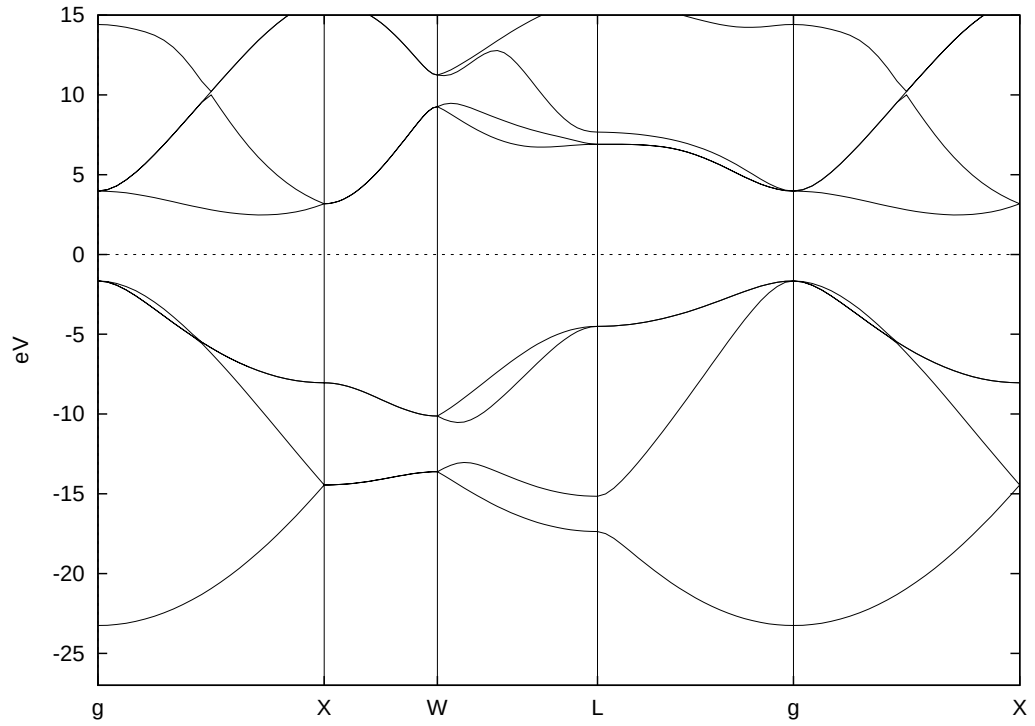


図 12: 炭素ダイヤモンドのバンド分散。入力ファイルは「work」ディレクトリ内の「Cdia.dat」。

次に OpenMX を以下の様に、通常実行します。

```
% ./openmx Cdia.dat
```

ここでは単一コアを使用した計算例を示しましたが、もちろん並列計算も実行可能です。この実行が正常に終了すると、「work」ディレクトリに「cdia.Band」というファイルが生成されます。キーワード「Band.KPath.UnitCell」が存在する場合にはこの単位胞ベクトルを用いて **k** 点経路の指定における逆格子ベクトル基底が計算されます。もしこのキーワード「Band.KPath.UnitCell」が指定されない場合には、「Atoms.UnitVectors」による単位胞ベクトルを用いて逆格子ベクトルを計算し、その目的に使用されます。

(2) データを gnuplot 形式に変換する

「source」ディレクトリに「bandgnu13.c」というファイルがあります。このファイルを次のようにしてコンパイルします。

```
% gcc bandgnu13.c -lm -o bandgnu13
```

コンパイルが正常に終了すると、「source」ディレクトリに実行可能なファイル「bandgnu13」が生成します。このファイルを「work」ディレクトリにコピーして下さい。実行ファイル「bandgnu13」を使い、次のコマンドを実行することにより、ファイル「cdia.Band」は gnuplot 形式に変換されます。

```
% ./bandgnu13 cdia.Band
```

これで、「cdia.GNUBAND」、「cdia.BANDDAT1」（および「cdia.BANDDAT2」）の 2 または 3 個のファイルが生成されます。ファイル「cdia.GNUBAND」は gnuplot のスクリプトであり、アップとダウンのスピン状態のデータファイル（それぞれ「cdia.BANDDAT1」および「cdiaBANDDAT2」）を読み込むためのファイルです。もし「LSDA-CA」、「LSDA-PW」、「GGA-PBE」のいずれかを用いるスピン分極計算を行った場合には、「*.BANDDAT1」に加えてダウンスピン状態の「*.BANDDAT2」が生成されます。ファイル「cdia.GNUBAND」は次のコマンドにより、gnuplot を使ってプロットすることができます。

```
% gnuplot cdia.GNUBAND
```

図 12 はここで説明した方法で計算した炭素ダイヤモンドのバンド分散を示しています。「cdia.GNUBAND」内の y 軸の範囲は変更されています。化学ポテンシャルがエネルギーの原点となるように自動的に移動されていますので、注意して下さい。化学ポテンシャルの具体的な値は、「cdia.out」中に記載されています。

逆格子の基底ベクトルを適切に選ぶと、バンド分散の計算において **k** 点の指定が容易になります。バンド分散の計算で使用する単位胞ベクトルを新たに設定する場合には、「Band.KPath.UnitCell」キーワード」を用いて、下記の例のように指定します。

```
<Band.KPath.UnitCell
 3.56 0.0 0.0
 0.0 3.56 0.0
 0.0 0.0 3.56
Band.KPath.UnitCell>
```

記述は「<Band.KPath.UnitCell」で開始し、「Band.KPath.UnitCell>」で終了します。
「Band.KPath.UnitCell」キーワードの記述がある場合、同キーワードで指定した単位胞ベクトルからバンド分散の計算に使用する逆格子基底ベクトルを算出します。「Band.KPath.UnitCell」を指定しない場合には、「Atoms.UnitVectors」キーワードで指定した単位胞ベクトルから計算した逆格子ベクトルが基底として使用されます。面心立方格子構造（fcc）や体心立方格子構造（bcc）の場合には「Band.KPath.UnitCell」キーワードで逆格子ベクトルを再定義した方が、特殊 **k** 点の指定が容易です。

18　状態密度

18.1　通常の方法

Kohn-Sham 固有値の状態密度（DOS）は次の 2 つの段階を経由して、計算されます。

(1) SCF 計算

ダイヤモンド構造の炭素を例として、DOS の計算方法を説明します。「work」ディレクトリ内のファイル「Cdia.dat」には、次の様に DOS 計算のためのキーワードが指定されています。

```
Dos.fileout                 on
Dos.Erange              -25.0  20.0
Dos.Kgrid                12 12 12
```

キーワード「Dos.Erange」で指定された最初と 2 番目の数値は、それぞれ DOS 計算のためのエネルギー範囲（eV）の下限と上限であり、エネルギーの原点（0.0）は化学ポテンシャルに対応します。またキーワード「Dos.Kgrid」で指定された数値（n1,n2,n3）は、DOS 計算での第 1 ブリルアンゾーンを離散化するためのグリッド数です。

次に OpenMX を以下の様に、通常実行します。

```
% ./openmx Cdia.dat
```

ここでは単一コアを使用した計算例を示しましたが、もちろん並列計算も実行可能です。計算が正常に終了すると、「work」ディレクトリに「cdia.Dos.val」および「cdia.Dos.vec」の 2 つのファイルが生成されます。「cdia.Dos.val」にはテキスト形式で固有値が、「cdia.Dos.vec」にはバイナリ形式で固有ベクトルが保存されています。この DOS 計算は、O(N) 計算にも対応しており、この場合にはガウシアンブロードニング法（Gaussian broadening method）が適用されます。

(2) DOS 計算

DOS 計算用のプログラムパッケージをコンパイルして下さい。「source」ディレクトリ内で、次のコマンドでコンパイルします。

```
% make DosMain
```

コンパイルが正常に終了すると、「source」ディレクトリ内に実行ファイル「DosMain」が生成されます。「DosMain」を「work」ディレクトリにコピーして、「work」ディレクトリに移動してください。このプログラム「DosMain」を使って先の 2 個のファイル「cdia.Dos.val」および「cdia.Dos.vec」から全状態密度 (DOS) と射影した DOS(PDOS) を次の様に計算します。

```
% ./DosMain cdia.Dos.val cdia.Dos.vec
```

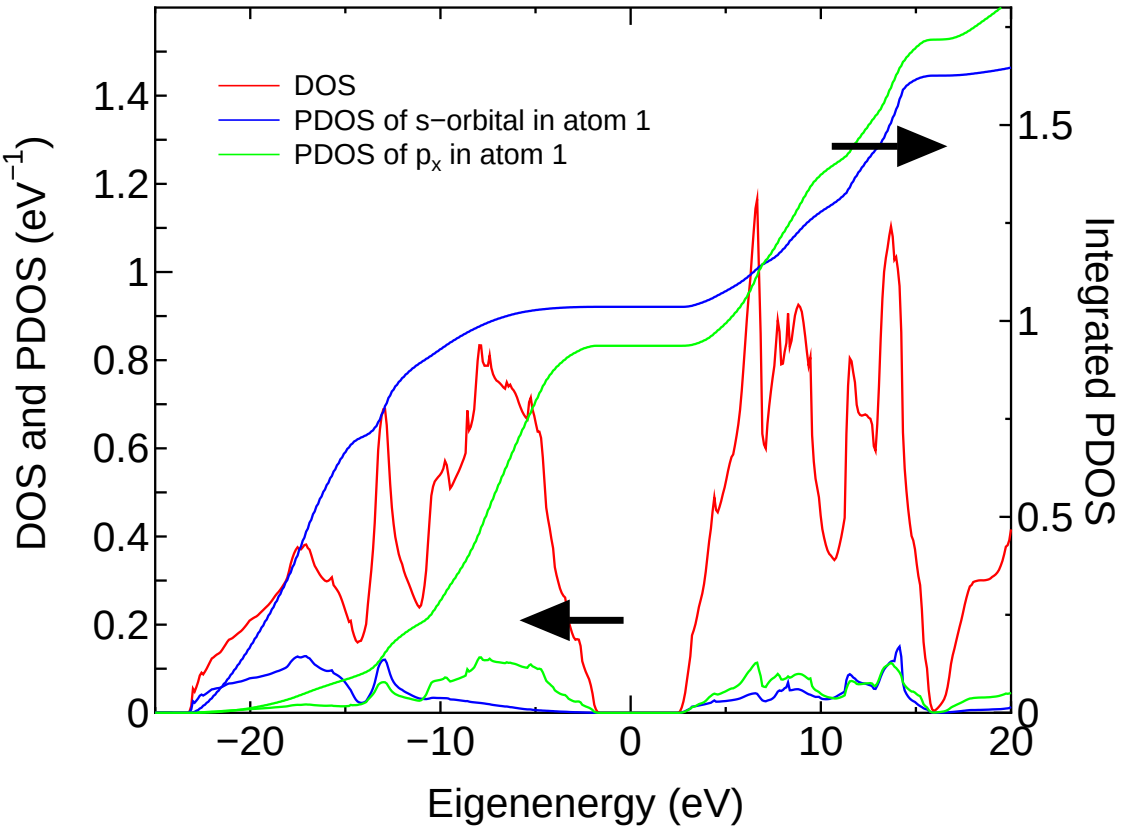


図 13: ダイヤモンド炭素の全状態密度 (DOS) と部分状態密度 (PDOS)、また PDOS の積分曲線。フェルミ準位をゼロに設定。s、 p、d 軌道の間で電荷の再配分が起っているため、s および p 軌道のフェルミ準位までの PDOS の積分値は正確には 1 ではないことに注意。「work」ディレクトリにある入力ファイル「Cdia.dat」を用いて実行。

その際に、プログラムから対話形式で次のように質問されることでしょう。

```
% ./DosMain cdia.Dos.val cdia.Dos.vec
Max of Spe_Total_CNO = 8
1 1 101 102 103 101 102 103
<cdia.Dos.val>
<cdia>
Which method do you use?, Tetrahedron(1), Gaussian Broadeninig(2)
1
Do you want Dos(1) or PDos(2)?
2

Number of atoms=2
Which atoms for PDOS : (1,...,2), ex 1 2
1
pdos_n=1
1
```

```
<Spectra_Tetrahedron> start
Spe_Num_Relation 0 0 1
Spe_Num_Relation 0 1 1
Spe_Num_Relation 0 2 101
Spe_Num_Relation 0 3 102
Spe_Num_Relation 0 4 103
Spe_Num_Relation 0 5 101
Spe_Num_Relation 0 6 102
Spe_Num_Relation 0 7 103
make cdia.PDOS.Tetrahedron.atom1.s1
make cdia.PDOS.Tetrahedron.atom1.p1
make cdia.PDOS.Tetrahedron.atom1.p2
make cdia.PDOS.Tetrahedron.atom1.p3
make cdia.PDOS.Tetrahedron.atom1
```

DOS の計算には四面体法（tetrahedron method）[48] もしくはガウシアンブロードニング法（Gaussian broadening method）が選択できます。またユーザーは DOS または PDOS を選ぶことができます。PDOS の計算を選択する場合は、PDOS を評価する原子を選択してください。この場合には、選択した原子の軌道（s, px(p1), py(p2), pz(p3),..）上に射影した PDOS が各ファイルに出力されます。これらのファイルでは、最初と二番目の列は、エネルギー（eV）、および DOS（eV^{-1}）または PDOS（eV^{-1}）の値であり、3 番目の列の数値は DOS または PDOS の積分値を示します。「LSDA-CA」、「LSDA-PW」、または「GGA-PBE」を用いたスピン分極計算を行った場合、これらのファイルの第 2 列および第 3 列の数値は、それぞれアップスピン状態およびダウンスピン状態の DOS と PDOS に対応し、第 4 列と第 5 列の数値はそれぞれに対応する積分値です。ガウシアンブロードニング法を使用した場合、ガウス分布のパラメーター値、a（eV）($\exp(-(E/a)^2)$ で定義されるガウス分布の幅を決めるパラメーター) を設定しなければなりません。ここで説明した手続きによって得られたダイヤモンド炭素の DOS および PDOS を図 13 に示します。

18.2 多数の k 点で計算する場合

多数の k 点を伴なう大規模系の状態密度（DOS）計算では大きなサイズのメモリーが必要になり、「DosMain」を用いた後処理計算において、メモリ不足のために計算が異常終了してしまうことがあります。このような場合、on-the-fly のガウシアンブロードニング法が利用できます。この方法では、OpenMX の DOS 計算の際に、on-the-fly でガウシアンブロードニング法が適用され、DOS の計算が行われます。大きなデータ量となる波動関数の情報はファイル「*.DOS.vec」に記録されません。この方法では、大容量のメモリを必要としませんので、大規模な系の状態密度計算に可能です。この方法を用いる場合には、入力ファイルに次の様にキーワードを指定します。

```
DosGauss.fileout      on       # default=off, on|off
DosGauss.Num.Mesh    200       # default=200
```

```
DosGauss.Width        0.2        # default=0.2 (eV)
```

この方法を使用する場合、キーワード「DosGauss.fileout」を「on」と指定します。そして、キーワード「DosGauss.Num.Mesh」では、キーワード「Dos.Erange」で指定したエネルギー範囲の分割数を指定します。キーワード「DosGauss.width」では、ガウス分布の $\exp(-(E/a)^2)$ を定義する幅 a を指定します。キーワード「DosGauss.fileout」と「Dos.fileout」は排他的で併用することは出来ませんので、注意して下さい。従って、このキーワードを使用する場合には、「Dos.fileout」は、次の様に「off」に設定します。

```
Dos.fileout           off        # on|off, default=off
```

また「Dos.fileout」および「DosGauss.file」の両方のキーワードに対して次の 2 つのキーワードが有効です。

```
Dos.Erange        -20.0  20.0    # default=-20 20
Dos.Kgrid            5 5 5       # default=Kgrid1 Kgrid2 Kgrid3
```

キーワード「DosGauss.fileout」では、ガウシアンブロードニング法による DOS だけしか計算出来ません。四面体法による DOS は計算できませんので、注意して下さい。「DosGauss.fileout」を用いた場合でも、「DosMain」による後処理の手続きは、四面体法による DOS が計算不可であること以外は、これまで説明した方法と同様です。

19　軌道の最適化

OpenMX で用いられる基底の動径関数は、軌道最適化法 [28] を用いて変分的に最適化することができます。メタン分子（入力ファイル「Methane_OO.dat」）を例として、軌道最適化の手順を説明します。軌道最適化法では、最適化動径関数は、プリミティブ動径関数の線形結合で表されます。線形結合の係数は縮約係数と呼ばれ、この縮約係数が変分原理に基づき最適化されます。軌道最適化法でのプリミティブ動径関数と最適化動径関数の数は、次の様にして指定します。

```
<Definition.of.Atomic.Species
  H    H5.0-s4>1            H_CA13
  C    C5.0-s4>1p4>1        C_CA13
Definition.of.Atomic.Species>
```

水素原子（H）の s 軌道に対しては、４つのプリミティブ動径関数の線形結合から１つの最適化動径関数が得られます。同様に、炭素原子（C）の場合、４つのプリミティブ動径関数の線形結合から１つの s(p) 軌道の最適化動径関数が得られます。さらに、以下のキーワードが軌道最適化法に関連します。

```
orbitalOpt.Method          species     # Off|Species|Atoms
orbitalOpt.Opt.Method        EF        # DIIS|EF
orbitalOpt.SD.step         0.001       # default=0.001
orbitalOpt.HistoryPulay     30         # default=15
orbitalOpt.StartPulay       10         # default=1
orbitalOpt.scf.maxIter      60         # default=40
orbitalOpt.Opt.maxIter     140         # default=100
orbitalOpt.per.MDIter       20         # default=1000000
orbitalOpt.criterion       1.0e-4      # default=1.0e-4

CntOrb.fileout               on        # on|off, default=off
Num.CntOrb.Atoms             2         # default=1
<Atoms.Cont.Orbitals
 1
 2
Atoms.Cont.Orbitals>
```

入力ファイル「Methane_OO.dat」を用いて OpenMX を通常実行します。

```
% ./openmx Methane_OO.dat
```

この計算が正常に終了すると、ファイル「met_oo.out」に軌道最適化の履歴が記録されています。

```
***********************************************************
***********************************************************
```

```
         History of orbital optimization   MD= 1
*********     Gradient Norm ((Hartree/borh)^2)     ********
              Required criterion=  0.000100000000
*************************************************************

   iter=   1  Gradient Norm=  0.057098961101  Uele= -3.217161102876
   iter=   2  Gradient Norm=  0.044668461503  Uele= -3.220120116009
   iter=   3  Gradient Norm=  0.034308306321  Uele= -3.223123238394
   iter=   4  Gradient Norm=  0.025847573248  Uele= -3.226177980300
   iter=   5  Gradient Norm=  0.019106400842  Uele= -3.229294858054
   iter=   6  Gradient Norm=  0.013893824906  Uele= -3.232489198284
   iter=   7  Gradient Norm=  0.010499500005  Uele= -3.235304178159
   iter=   8  Gradient Norm=  0.008362635043  Uele= -3.237652870812
   iter=   9  Gradient Norm=  0.006959703539  Uele= -3.239618540761
   iter=  10  Gradient Norm=  0.005994816379  Uele= -3.241268535418
   iter=  11  Gradient Norm=  0.005298095979  Uele= -3.242657118263
   iter=  12  Gradient Norm=  0.003059655878  Uele= -3.250892948269
   iter=  13  Gradient Norm=  0.001390201488  Uele= -3.255123241210
   iter=  14  Gradient Norm=  0.000780925380  Uele= -3.255179362845
   iter=  15  Gradient Norm=  0.000726631072  Uele= -3.255263012792
   iter=  16  Gradient Norm=  0.000390930576  Uele= -3.250873416989
   iter=  17  Gradient Norm=  0.000280785975  Uele= -3.250333677139
   iter=  18  Gradient Norm=  0.000200668585  Uele= -3.252345643243
   iter=  19  Gradient Norm=  0.000240367596  Uele= -3.254238199726
   iter=  20  Gradient Norm=  0.000081974594  Uele= -3.258146794679
```

多くの場合、20〜50 回の反復計算で収束に達します。プリミティブ基底関数および最適化基底関数で計算されたメタン分子の全エネルギーの比較結果を以下に示します。

```
    Primitive basis orbitals
       Utot  =      -7.992569945749 (Hartree)

    Optimized orbitals by the orbital optimization
       Utot  =      -8.133746986502 (Hartree)
```

軌道最適化によって、少ない基底数で高精度な基底関数が得られることが分かります。図 14 に、プリミティブ基底および最適化基底を使って得られた分子およびバルクの全エネルギーの収束の様子を示します。ここで扱った全ての系に対して、最適化基底の収束特性が優れていることが分かります。上記の例でのメタン分子の場合、最適化された基底関数は「C_1.pao」と「H_2.pao」の 2 つのファイルに出力さ

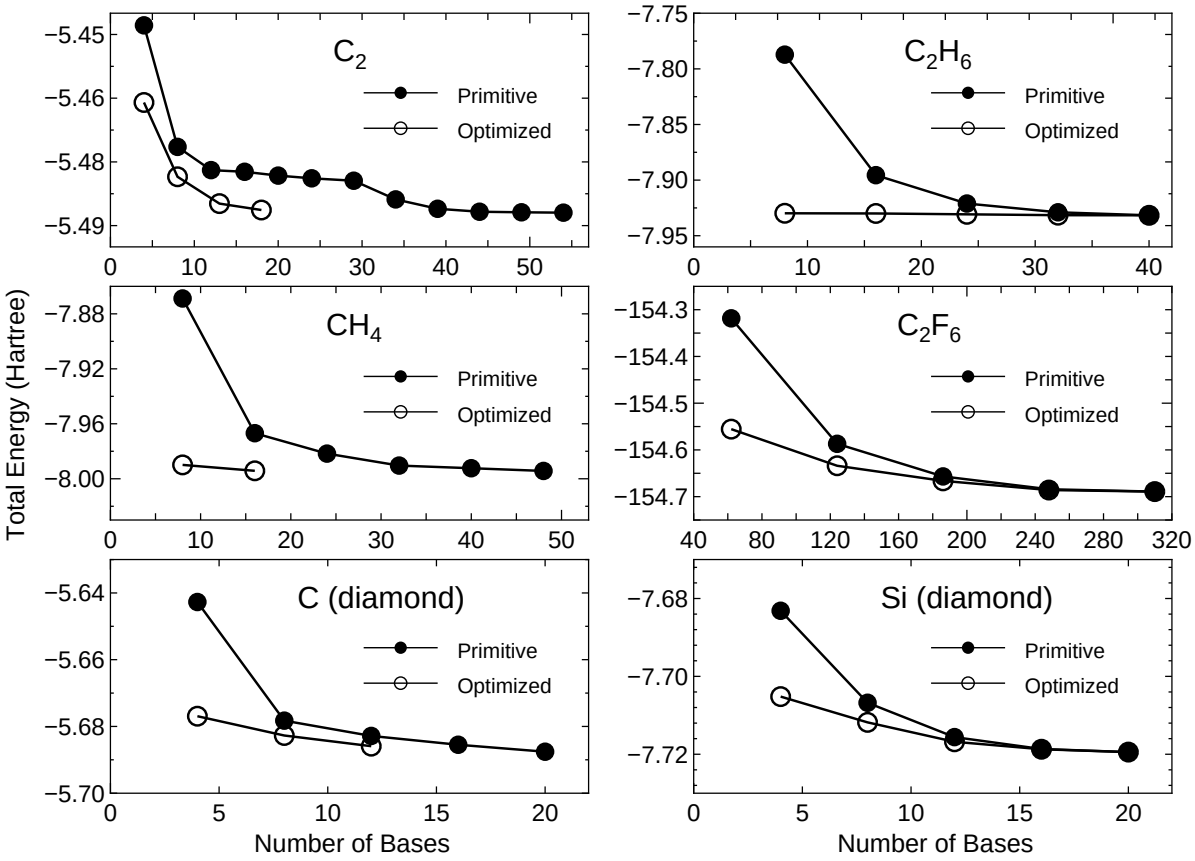


図 14: プリミティブ基底関数および最適化基底関数で計算された炭素２量体（C_2）、メタン分子（CH_4）、ダイヤモンド構造の炭素とケイ素、エタン分子（C_2H_6）、ヘキサフルオロエタン（C_2F_6）の全エネルギー。全エネルギーと全基底数は、C_2、ダイヤモンド構造の炭素およびケイ素の場合は１原子についての値、CH_4、C_2H_6 および C_2F_6 の場合は１分子についての値。

れます。これらのファイル「C_1.pao」と「H_2.pao」は擬原子基底関数の入力データとして、OpenMX の計算に対してそのまま使用できます。最適化された基底関数はファイルに出力されますので、計算しようとする系の基底関数を予め最適化しておくと便利です。この際、軌道最適化法を適用する系としては、化学的に類似した小さな系を選択することを推奨します。

キーワード「orbitalOpt.Method」には、次の２つのオプションが用意されています。(1) それぞれの原子上の基底関数が完全に最適化される「atoms」、(2) それぞれの原子種の基底関数が最適化された「species」。

- atoms

  各原子単位で基底関数が最適化されます。動径波動関数 R は磁気量子数には依存しませんので、系の回転に対する全エネルギーの不変性が保証されます。

- species

  「Definition.of.Atomic.Species」で定義した同一名の原子種の基底関数は同一の軌道に最適化されます。動径波動関数 R は磁気量子数には依存しませんので、系の回転に対する全エネルギーの不

変性が保証されます。ほぼ同様の化学的環境を持つ原子に対して同一の最適化基底を生成したい場合に、有用な方法です。

「入力ファイル」の章でも同様な情報が記載されていますが、ユーザーの利便性のため、関連するキーワードの詳細を以下に列挙します。

orbitalOpt.scf.maxIter
軌道最適化における SCF 反復の最大回数を「orbitalOpt.scf.maxIter」キーワードで指定します。

orbitalOpt.Opt.maxIter
軌道最適化の反復の最大回数を「orbitalOpt.Opt.maxIter」キーワードで指定します。軌道最適化の反復は、収束条件が達成しなかった場合でも、同キーワードで設定した回数で終了します。

orbitalOpt.Opt.Method
軌道最適化の収束方法として、2 つの手法がサポートされています。「EF」は固有ベクトル追跡法、「DIIS」は反復部分空間における直接反転法です。それぞれのアルゴリズムは構造最適化のそれと同じです。「orbitalOpt.Opt.Method」キーワードでは「EF」あるいは「DIIS」を指定してください。

orbitalOpt.StartPulay
「orbitalOpt.StartPulay」キーワードで指定した最適化ステップから、準ニュートン法である「EF」または「DIIS」法を開始します。

orbitalOpt.HistoryPulay
準ニュートン法である「EF」および「DIIS」法において、次ステップでの縮約係数を推定するために参照する過去のステップ数を「orbitalOpt.HistoryPulay」キーワードで指定します。

orbitalOpt.SD.step
準ニュートン法である「EF」および「DIIS」法を開始するまでの最適化ステップは最急降下法が適用されます。最急降下法で使用する前因子は「orbitalOpt.SD.step」キーワードで指定します。多くのケースにおいて、「orbitalOpt.SD.step」の適切な値は 0.001 程度となります。

orbitalOpt.criterion
軌道最適化の収束条件（$(\text{Hartree/bohr})^2$）を「orbitalOpt.criterion」キーワードで指定します。「微分のノルム <orbitalOpt.criterion」という条件が満たされた時に反復ループが終了します。

CntOrb.fileout
最適化動径関数をファイルに出力したい場合は、「CntOrb.fileout」キーワードを「ON」にする必要があります。

Num.CntOrb.Atoms
最適化動径関数をファイルに出力する際の原子数を「Num.CntOrb.Atoms」キーワードで指定します。

Atoms.Cont.Orbitals
「Atoms.SpeciesAndCoordinates」キーワードの第 1 列で定義した原子の通し番号を用いて、最適化基底を出力する原子を「Atoms.Cont.Orbitals」キーワードで次のように指定します。

```
<Atoms.Cont.Orbitals
 1
 2
Atoms.Cont.Orbitals>
```

記述は「<Atoms.Cont.Orbitals」で始め、「Atoms.Cont.Orbitals>」で終わります。行の数は、「Atoms.Cont.Orbitals」で記述する数字と整合性がなければなりません。例えば、最適化擬原子軌道が「C_1.pao」と「H_2.pao」などとして保存された場合、その原子種名はキーワード「Definition.of.Atomic.Species」の設定における第一列の記号に対応し、記号の後の数字はキーワード「Atoms.SpeciesAndCoordinates」の設定における第一列の数値を意味します。これらの出力ファイル「C_1.pao」と「H_2.pao」は基底関数の入力データとして使用可能です。

20　O(N) 法

一般に行列対角化の計算時間は、使用する基底関数の数の 3 乗に比例します。そのため大規模な系の計算では、対角化が計算の律速となります。一方、近似手法である O(N) 法を使うと計算量は O(N) となり、大規模系の取り扱いが可能となります。しかし計算精度に関しては注意深い配慮が必要となります。OpenMX Ver. 3.7 では、分割統治法 (DC 法) [37] と O(N) Krylov 部分空間法 [30] の二つの O(N) 法が利用可能です。以下の節では、それぞれの O(N) 法について計算例を示しながら説明します。

20.1　分割統治法 (DC 法)

DC 法は数値的に安定な方法で広範囲の物質に適用可能であり、また計算精度と計算効率を容易に制御できる方法です。本手法は特に共有結合性の物質群に適しています。この節では、DC 法を用いた O(N) 計算について説明します。「work」ディレクトリ内の入力ファイル「DIA8_DC.dat」において、キーワード「EigenvalueSolver」を使って DC を指定して下さい。

```
scf.EigenvalueSolver    DC
```

入力ファイル「DIA8_DC.dat」を用いて OpenMX を通常実行します。

```
% ./openmx DIA8_DC.dat
```

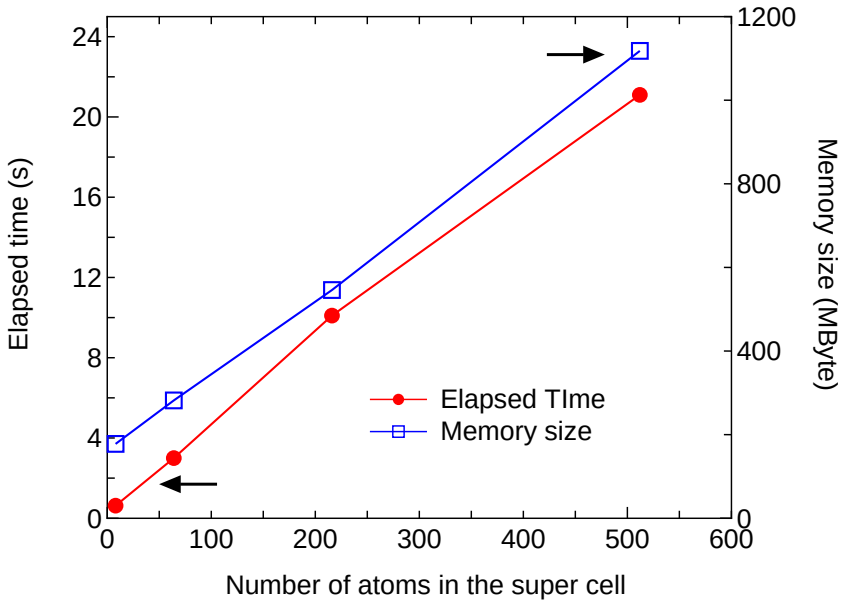


図 15: 分割統治法によるダイヤモンド炭素の計算時間とメモリ使用量。計算時間は 1 回の SCF ステップに対する対角化の所要時間、またメモリ使用量は 1 つの MPI プロセス当りを示す。計算には MPI 並列計算で 16 プロセスを使用。基底関数には C5.0-s1p1、DC 法では orderN.HoppingRanges=6.0 (Å) を使用。使用計算機は Xeon プロセッサ（2.6 GHz)。計算に用いた入力ファイルは「work」ディレクトリ内の「DIA8_DC.dat」、「DIA64_DC.dat」、「DIA216_DC.dat」、「DIA512_DC.dat」。

表 2: 通常の対角化法と DC 法によって計算された C_{60} 分子、小ペプチド分子（バロルフィン (valorphin) [63]）、シトシンとグアニンから構成される DNA の全エネルギーと計算時間。計算には最小基底関数を使用。DC の後のカッコ内の数値は、DC 計算に使われた「orderN.HoppingRanges」を示す。計算時間は Opteron PC クラスター（48 MPI プロセス, 2.4 GHz）用いて測定。入力ファイルは、「work」ディレクトリ内の「C60_DC.dat」、「Valorphin_DC.dat」、「CG15c_DC.dat」。

	Total energy (Hartree)	Computational time (s)
C_{60}		
(60 atoms, 240 orbitals)		
Conventional	-343.89680	36
DC (7.0)	-343.89555	37
Valorphin		
(125 atoms, 317 orbitals)		
Conventional	-555.28953	81
DC (6.5)	-555.29019	76
DNA		
(650 atoms, 1880 orbitals)		
Conventional	-4090.95463	576
DC (6.3)	-4090.95092	415

この入力ファイルは 8 個の炭素原子を含むダイヤモンド格子の DC 計算を行うためのものです。構造最適化は行いません。計算時間は Xeon プロセッサ （2.6 GHz）を使用した場合にはおよそ 120 秒です。図 15 に、分割統治法によるダイヤモンド炭素の計算時間と計算メモリ量をスーパーセル内の炭素原子数の関数として示します。計算時間は１回の SCF ステップに対する対角化の所要時間、またメモリ使用量は１つの MPI プロセス当りを示します。図から、計算時間とメモリ使用量は原子数にほぼ比例することが分かります。DC 法の計算精度と計算効率はキーワード「orderN.HoppingRanges」で制御されます。

- orderN.HoppingRanges

  各原子を中心とする球の半径 (Å) を「orderN.HoppingRanges」キーワードで定義します。DC 法および O(N) クリロフ部分空間法では、この球内に含まれる原子を選択することで切り取られたクラスタが構成されます。

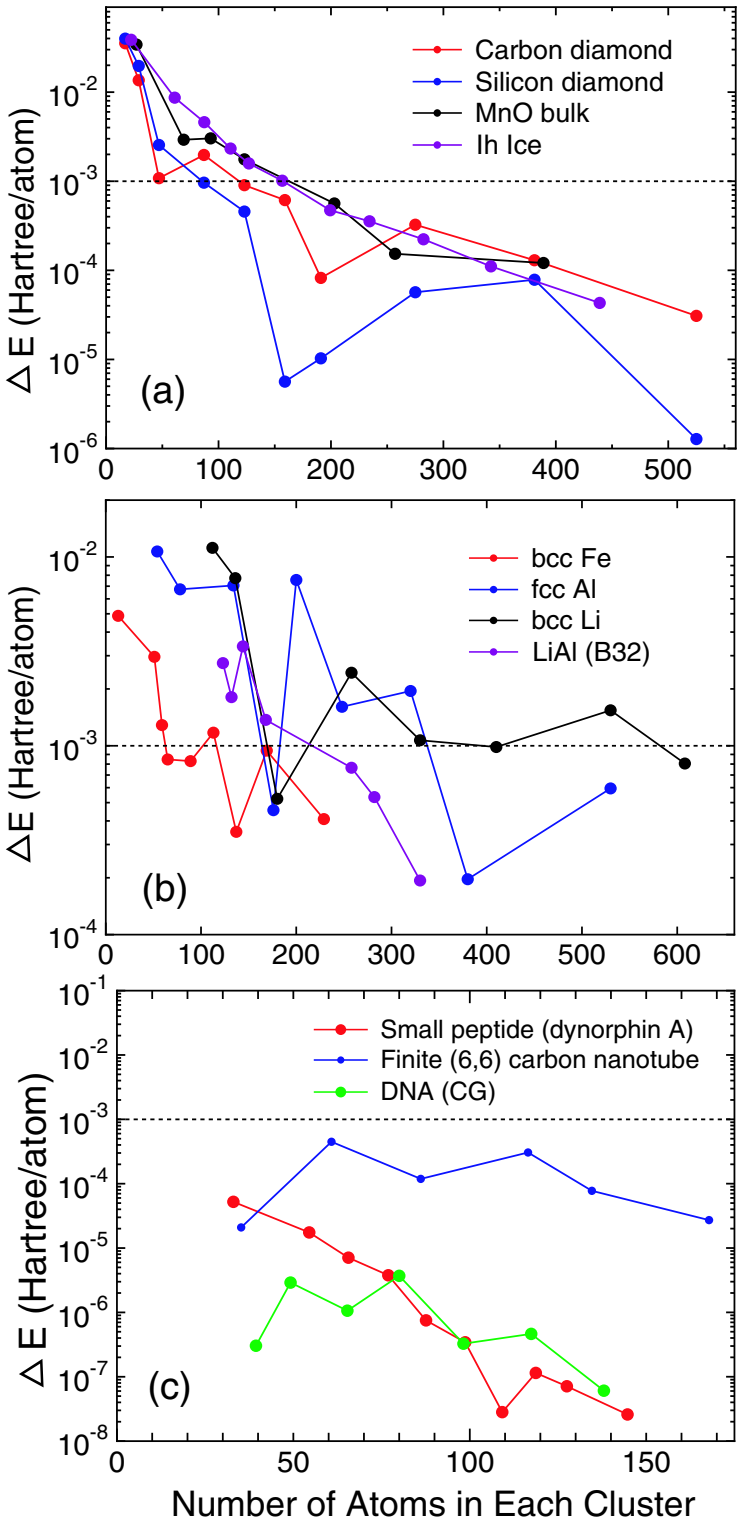


図 16: DC 法で計算された (a) 有限ギャップを持った固体、(b) 金属、 (c) 分子系の全エネルギーにおける誤差。横軸は各切り取られたクラスタ内の原子数。水平方向の点線は「ミリハートリー（mili-Hartree)」の精度を示す。

系の原子数を N とすると、「orderN.HoppingRanges」で指定された半径の球で物理的に切り取られたクラスターが N 個、構築されます。それぞれのクラスターに対して、独立に固有値問題を解き、中心原子への射影状態密度を計算します。その後、全ての原子からの射影状態密度を足し合せることで、全系の状態密度が計算されます。「orderN.HoppingRanges」の適切な値は系によって異なりますが、分子系では計算精度と計算効率の妥協点として次の値が推奨されます。

```
orderN.HoppingRanges      6.0 - 7.0     # in Ang.
```

表 2 に、C_{60} 分子と小ペプチド分子（バロルフィン (valorphin) [63]）およびシトシンとグアニンから構成される DNA について、通常の対角化法と DC 法で計算した全エネルギーの比較を示します。DC 法で計算した全エネルギーの誤差は、どの系でも 1 原子当りミリハートリー程度であることが分かります。また本計算で用いた計算条件では原子数が 500 以上になると DC 法が通常の対角化法より高速となることが推定されますが、計算時間における通常の対角化法と DC 法の交差点は、系と並列計算で使用するコア数に依存しています。

切り取られたクラスターのサイズによって全エネルギーの収束性がだいたいどのように変わるかを見るため、通常の対角化と比較した全エネルギーの誤差を図 16 に示します。ここでは切り取られたクラスターの原子数をパラメーターとして (a) 有限ギャップを持った固体、(b) 金属、(c) 分子の 3 つの場合を取り上げました。これから分かることは、(a) 有限ギャップを持った固体と (c) 分子系の場合にはその誤差がほぼ指数関数的に減少し、(b) 金属の場合には収束速度が相対的に遅いということです。

20.2 O(N)Krylov 部分空間法

DC 法は広範な系に適用でき、数値的にも安定な方法です。しかし金属系に対して精確な結果を得るには、図 16 に示したように切り取られたクラスターのサイズを大きくする必要があり、その場合には計算量が増大します。計算量を削減する一つの方法は、切り取られたクラスターによって定義されるベクトル空間を次元数がより小さな Krylov 部分空間 [30] にマッピングすることです。次のキーワードにより O(N)Krylov 部分空間法を利用できます。

```
scf.EigenvalueSolver        Krylov
```

基本的に計算精度と計算効率は次の 2 つのキーワードで制御されます。

```
orderN.HoppingRanges          6.0
orderN.KrylovH.order          400
```

キーワード「orderN.HoppingRanges」は、DC 法の場合と同様に、各原子を中心とする球の半径を定義します。ハミルトニアンに対する Krylov 部分空間の次元は、キーワード「orderN.Krylov.order」で指定します。さらに次のキーワードで O(N)Krylov 部分空間法による計算の詳細が設定可能です。

- orderN.Exact.Inverse.S　　on| off, default=on

  キーワード「orderN.Exact.Inverse.S」を「on」に設定すると、切り取られた各クラスタの重なり行列の逆行列は厳密に評価されます。off に設定する場合はキーワード「orderN.KrylovS.order」の項を参照してください。デフォルトでは「on」に設定されています。

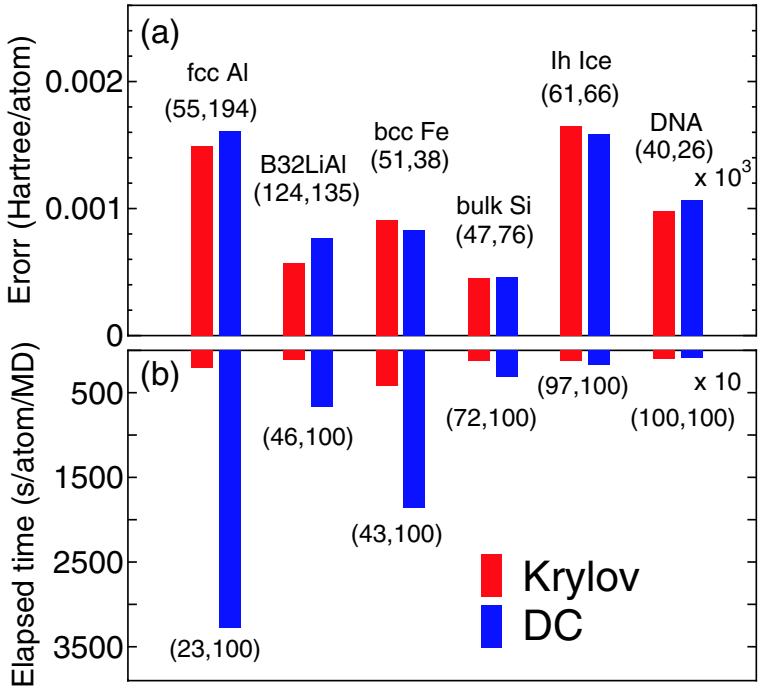


図 17: (a) Krylov 部分空間法と DC 法で計算された金属および絶縁固体の全エネルギー（Hartree/atom）における絶対誤差と (b) その計算時間（s/atom/MD）。比較を容易にするため、1 MPI プロセスを用いて計算。図 (a) のカッコ内の数値はコア領域およびバッファ領域における平均原子数。図 (b) のカッコ内の数値は、切り取られたクラスター内の基底関数の数に対する Krylov 部分空間の次元の百分率を示す。

- orderN.KrylovS.order　　1200, default=orderN.KrylovH.order×4

  「orderN.Exact.Inverse.S=off」の場合、Krylov 部分空間法によって逆行列を近似します。この時、切り取られた各クラスタに対する重なり行列のクリロフ部分空間法の次元を「orderN.KrylovS.order」キーワードで指定します。デフォルト値は、「orderN.KrylovH.order」で設定した値の 4 倍です。

- orderN.Recalc.Buffer　　on| off, default=on

  キーワード「orderN.Recalc.Buffer」を「on」に設定すると、バッファ行列は各 SCF 反復毎に再計算されます。「off」の場合にはバッファ行列は初回の SCF ステップで計算され、その後の SCF 反復では固定されます。デフォルトでは「on」に設定されています。

- orderN.Expand.Core　　on| off, default=on

  キーワード「orderN.Expand.Core」を「on」に設定すると、コア領域は半径 $1.2 \times r_{\min}$ の球の中にある原子から構成されます。ここで $r_{\min}$ は、中心原子と最隣接原子間の距離です。このコア領域はクリロフ部分空間を生成するときの第 1 ステップで使用されるベクトル群を定義します。「orderN.Expand.Core」が「off」の場合、中心の原子がコア領域と見なされます。デフォルトでは「on」に設定されています。

一般にキーワード「orderN.Exact.Inverse.S」と「orderN.Expand.Core」は共有結合性物質に対しては「on」とした方が収束性が向上しますが、単純金属では「off」とした方が収束性が向上する場合もあり

ます。図 17 に、種々の物質に対し、Krylov 法と DC 法で計算した全エネルギーの絶対誤差を示します。特に金属系では DC 法と比較して Krylov 部分空間法の方が効率的であり、共有結合性とイオン性の増大とともに計算時間は同じ程度になることが分かります。

大規模な計算を実現するために、O(N) Krylov 部分空間法は並列化に配慮して実装されています。MPI による並列化の場合には原子数と同じ MPI プロセス数までは計算を加速させることが出来ます。また OpenMP/MPI ハイブリッド並列によって OpenMP スレッドを使用すればさらに計算速度の向上が可能です。超並列計算機を用いて 10 万個以上の原子から構成される系の並列計算を実施した例を図 18 に示します [31, 32]。この例では、ベンチマーク系として 131072 個の炭素原子から構成されるダイヤモンドの計算を行い、131072 MPI プロセスを使用し、その並列化効率は 68%に達しています。

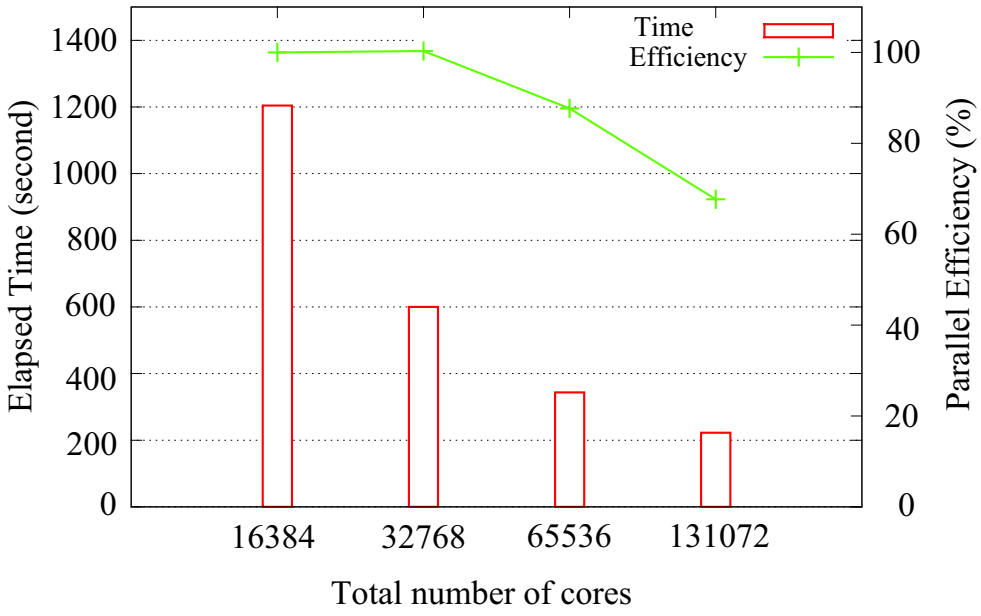


図 18: スーパーコンピュータ「京」上でのハイブリッド並列計算における O(N) Krylov 部分空間法の並列化効率。すべての場合で OpenMP の 8 スレッドを使用。ベンチマーク系として 131072 個の炭素原子から成るダイヤモンド構造を計算。

20.3 ユーザーによる FNAN+SNAN の定義

OpenMX Ver. 3.7 で実装されている O(N) 法では、切り取られたクラスター内の原子は**第 1 隣接**と**第 2 隣接**の 2 種類の原子に分類されます。中心原子上の基底関数のカットオフ半径 r_0 と隣接原子の基底関数のカットオフ半径 $r_{\rm N}$ の和 $(r_0 + r_{\rm N})$ が、原子間距離より小さい場合、その原子は第 1 隣接原子に分類され、この条件を満たさないクラスター内の原子は第 2 隣接原子に分類されます。第 2 隣接原子の選択は、キーワード「orderN.HoppingRanges」によって決定されており、第 1 と第 2 隣接原子数は、それぞれ **FNAN** および **SNAN** として標準出力に書き出されています。さらに「orderN.HoppingRanges」とは独立に SNAN が決定するキーワード「orderN.FNAN+SNAN」が利用可能です。このキーワードによって (FNAN+SNAN) の数値を以下の様にユーザーが直接、与えることが出来ます。

```
<orderN.FNAN+SNAN
```

```
  1  60
  2  65
  3  60
  4  50
  ..
  .
orderN.FNAN+SNAN>
```

この指定では、行数は原子数に等しくなければなりません。最初の列は、キーワード「Atoms.SpeciesAndCoordinates」の第 1 列と同一の通し番号であり、2 列目が (FNAN+SNAN) の値です。このキーワードを用いて、各原子に対して (FNAN+SNAN) が明示的に与えられた場合には、(FNAN+SNAN) の値が指定された値になるように各原子毎にクラスターの球半径が調整されます。キーワード「orderN.HoppingRanges」によってクラスターサイズを指定した場合の問題点は単位胞ベクトルを変化させた際に顕在化します。この場合、単位胞ベクトルを変化に伴い、大きく (FNAN+SNAN) が変化し、その結果としてエネルギー曲線上に大きな飛びが生じます。キーワード「orderN.FNAN+SNAN」によって明示的に (FNAN+SNAN) を与えることで、この様なケースを避けることができます。

21 MPI 並列化

大規模系の計算を実行するために、分散メモリ型並列計算機上での MPI 並列計算が実行可能です。

21.1 O(N) 計算

O(N) 法は並列化に適したアルゴリズム構造を持っているため、高い並列化効率が期待されます。典型的な MPI の実行は次の様に行います。

```
% mpirun -np 4 openmx DIA512_DC.dat > dia512_dc.std &
```

「work」ディレクトリ中の「DIA512_DC.dat」は、分割統治 (DC) 法を用いて 512 個の炭素原子からなるダイヤモンド格子の SCF 計算を実行するための入力ファイルです。MPI プロセス数に対する速度向上比を図 19(a) に示します。この並列計算は CRAY-XC30 （Xeon プロセッサ、2.6 GHz) 上で実行し、1MD ステップの経過時間から速度向上比が算出されました。128 個の MPI プロセスを用いて、その速度向上は約 84 倍であり、またプロセス数の増加に伴い、並列化効率は減少していくことが分かります。並列化効率が減少する理由は、計算負荷の大きな O(N) 対角化計算において 1 個のプロセスに割り当てられる原子数が減少し、disk I/O のような並列化されていない部分の経過時間が顕在化するためです。また原子種や幾何学構造が一様でない系では、負荷分散が崩れるために並列化効率が大きく減少する場合もあります。MD 計算や分子動力学計算では、動的に負荷分散を保つアルゴリズムが実装されており、可能な限り並列化効率の向上が図られています。並列化に関するさらなる情報については、「O(N)Krylov 部分空間法」の節を参照して下さい。

21.2 クラスター計算

クラスター計算では、スピン多重度と固有状態のループ構造に対して MPI 並列化が実装されています。スピン多重度は非スピン分極計算およびノンコリニア計算では 1、スピン分極計算では 2 となります。スピン多重度がまず優先的に並列化され、多数の MPI プロセスが使用された際には、さらに固有状態が並列化されます。OpenMX Ver. 3.7 では、クラスター計算の固有値問題は、高並列固有値ソルバー: ELPA [26] によって計算されます。図 19(b) に、Mn_{12} 単一分子磁石のスピン分極計算の所要時間と速度向上比を示します。使用した入力ファイル「Mn12.dat」は「work」ディレクトリ に保存されています。速度向上比は、32 プロセスと 64 プロセスで、それぞれ 11 および 17 であることが分かります。

21.3 バンド計算

バンド計算ではスピン多重度、**k** 点、固有状態の 3 つのループ構造に対して MPI 並列化が実装されています。スピン多重度は非スピン分極計算およびノンコリニア計算では 1、スピン分極計算では 2 となります。スピン多重度がまず優先的に並列化され、多数の MPI プロセスが使用された際には、さらに **k** 点、固有状態の順番で並列化されます。また MPI プロセス数が（スピン多重度）×（**k** 点数）以下の場

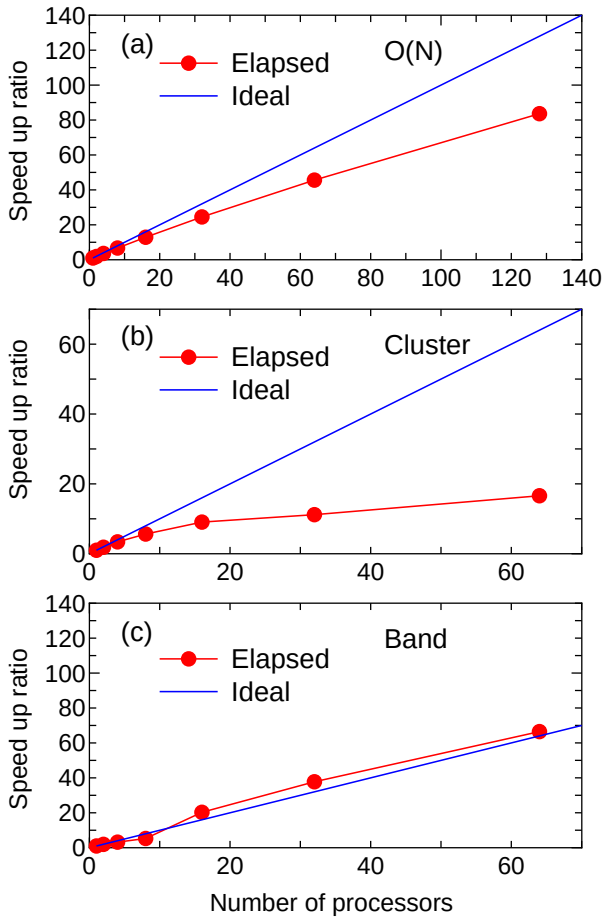


図 19: CRAY-XC30 （Xeon プロセッサ、2.6GHz）上で MPI 並列計算を行った場合の、1MD ステップの所要時間における速度向上比。(a) DC 法によるダイヤモンド (512 炭素原子) の計算、(b) クラスター法による Mn_{12} 単一分子磁石 (148 原子) の計算、(c) バンド法によるダイヤモンド (64 炭素原子、3×3×3 の k 点) の計算。比較のために、理想的な速度向上比に対応する直線も示す。

合、各 **k** 点で化学ポテンシャルと密度行列の各計算のために 2 回の対角化が行われます。これは波動関数を保持するためのメモリ使用量を削減するためで、1 回目の対角化で化学ポテンシャルを探索し、2 回目の対角化で波動関数を保持することなく、直接に密度行列を計算しています。一方、MPI プロセス数が（スピン多重度）×（**k** 点数）を越える場合、各プロセスは多数の **k** 点での波動関数をメモリ上に保持する必要がないため、対角化は一度だけで、計算効率が見かけの上で向上します。図 19 (c) に示す様に、プロセス数の関数として速度向上比を見た場合、バンド計算の並列化効率は良好であることが分かります。この計算は 3×3×3 個の **k** 点を考慮したダイヤモンド (64 炭素原子) の非スピン分極バンド計算を行ったもので、使用した入力ファイルは「work」ディレクトリ にある「DIA64_Band.dat」です。この場合のスピン多重度は 1 です。またコリニア計算ではブリルアンゾーンの半分の **k** 点のみが考慮されます。**a** 軸、 **b** 軸、**c** 軸に対する全ての **k** 点数が奇数の場合には、Γ 点が含まれますので、実際の計算が行われた **k** 点数は (3×3×3-1)/2+1=14 となります。プロセス数が 14 以上では、速度向上比が理想値を超えていますが、これは上記で説明したアルゴリズムの変更によるものです。クラスター計算と同様に、バンド計算における固有値問題の解法には高並列固有値ソルバー: ELPA [26] を使用しています。

21.4　完全な 3 次元並列化

OpenMX Ver. 3.6 以下のバージョンでは、単位胞の **a** 軸に対する 1 次元領域分割法によりデータを分散し、それに基づき MPI 並列化が実装されていました。一方、OpenMX Ver. 3.7 ではデータ分散のために、完全な 3 次元領域分割をサポートしています。したがって、並列化の際に均等な負荷分散を実現するために単位胞をどのように指定したら良いかについて、考慮する必要がありません。OpenMX Ver. 3.7 では、単位胞ベクトルの選択に依存することなく、ほぼ同等の並列化効率が得られます。

21.5　使用可能な MPI プロセス数

OpenMX Ver. 3.6 以下のバージョンでは、並列計算で使用できる MPI プロセス数の最大値は、系に含まれる原子数で制限されていました。OpenMX Ver. 3.7 ではこのような制限はありません。MPI プロセス数が原子数より大きい場合でも、可能な限り MPI 並列化が効率良く行われます。この機能は、**k** 点数が系の原子数より十分に大きい場合に特に有用となるでしょう。

22 OpenMP/MPI ハイブリッド並列化

OpenMP/MPI を用いたハイブリッド並列計算は次のコマンドで実行可能です。

```
% mpirun -np 32 openmx DIA512-1.dat -nt 4 > dia512-1.std &
```

「-nt」によって各 MPI プロセスに割り当られたスレッド数を与えます。もし「-nt」が指定されない場合は、スレッド数は 1 にセットされ、この場合には通常の MPI 並列化となります。OpenMX Ver. 3.7 の並列化は OpenMX Ver. 3.6 からは大きく変更されていますので、計算効率とメモリ使用の効率に関して、通常の MPI 並列化と比較して、ハイブリッド並列化が有効かどうかまだ十分なデータがありません。しかし予備的なベンチマーク計算では、ハイブリッド並列化は、メモリの利用に関しては効率が良く、一方で計算効率に関しては両者とも同程度であるという結果となっています。通常の並列化と OpenMP/MPI ハイブリッド並列化の適切な選択は、使用する計算機に依存していると考えられますので、ベンチマーク計算によって、確認されることを推奨します。

23 大規模計算

23.1 通常の固有値解法

数百個のプロセッサコアが利用可能であれば、通常の対角化法で 1000 原子から構成される系の構造最適化が可能です。1000 原子系のベンチマーク計算は次のように「runtestL2」によって実行できます。

```
% mpirun -np 128 openmx -runtestL2 -nt 4
```

OpenMX は 7 個のテスト計算を実行し、計算結果を「work/large2_example」に保存されている参照データと比較します。以下に示すのは、CRAY-XC30 上で 128 個の MPI プロセスと 4 個の OpenMP スレッドを用いて実行した「runtestL2」の結果です。

1	large2_example/C1000.dat	Elapsed time(s)= 1731.83	diff Utot= 0.000000002838	diff Force= 0.000000007504
2	large2_example/Fe1000.dat	Elapsed time(s)=21731.24	diff Utot= 0.000000010856	diff Force= 0.000000000580
3	large2_example/GRA1024.dat	Elapsed time(s)= 2245.67	diff Utot= 0.000000002291	diff Force= 0.000000015333
4	large2_example/Ih-Ice1200.dat	Elapsed time(s)= 952.84	diff Utot= 0.000000000031	diff Force= 0.000000000213
5	large2_example/Pt500.dat	Elapsed time(s)= 6831.16	diff Utot= 0.000000002285	diff Force= 0.000000004010
6	large2_example/R-TiO2-1050.dat	Elapsed time(s)= 2259.97	diff Utot= 0.000000000106	diff Force= 0.000000001249
7	large2_example/Si1000.dat	Elapsed time(s)= 1655.25	diff Utot= 0.000000001615	diff Force= 0.000000005764

Total elapsed time (s) 37407.95

これら全ての計算において、各原子の基底関数には価電子基底関数を 2 つ、分極基底関数を 1 つを割り当てていますので、十分な精度を持った計算となっています。「Pt500.dat」を除く、他の全ての系は 1000 原子以上から構成されています。ここでファイル名の最後の数字はそれぞれの系に含まれる原子数を示しています。上記の表に示された 1 MD ステップ当たりの経過時間は 30 分程度です。したがって、数百個のプロセッサコアが利用可能であれば 1000 原子から構成される系の構造最適化が可能であることが分かります。この計算に用いた入力ファイルと出力ファイルはディレクトリ「work/large2_example」に保存されています。出力ファイルから得られた情報に基づき、1 SCF ステップ当たりの経過時間をまとめたものを以下に示します。

No.	Input file	SCF steps	Elapsed time(s/SCF/spin)	Dimension
1	large2_example/C1000.dat	44	35	13000
2	large2_example/Fe1000.dat	384	30	13000
3	large2_example/GRA1024.dat	54	35	13312
4	large2_example/Ih-Ice1200.dat	41	18	9200
5	large2_example/Pt500.dat	171	35	12500
6	large2_example/R-TiO2-1050.dat	35	57	15750
7	large2_example/Si1000.dat	48	34	13000

Kohn-Sham ハミルトニアンの次元は 10000 のオーダで、SCF ステップ当たりの所要時間は全ての系に対して 40 秒程度です。全所要時間の差は、主に SCF 反復回数の差から生じていることが分かります。

23.2 O(N) 法と通常の固有値解法の組み合わせ

O(N) 法によって 1000 個以上の原子を含む大規模な系を取り扱うことが可能ですが、OpenMX で実行される O(N) 法では、波動関数に関する情報を得ることが出来ません。大規模系に対する波動関数と

対応する固有値を得るための簡便な方法は、最初に O(N) 法を用いて自己無撞着な電子密度を求め、次にこの自己無撞着な電子密度の下で、一度だけ通常の対角化計算を行うものです。この方法の例として、564 個の炭素原子で構成される多重連結したカーボンナノチューブ（MCCN）の大規模計算を示します。まず最初に、16 MPI プロセス (Xeon, 2.6GHz) と O(N) Krylov 部分空間法を用いて MCCN の SCF 計算を実行しました。計算条件は以下の通りです。C5.0-s2p1 （基底関数)、「scf.energycutoff=130 Ryd」、「scf.criterion=1.0e-7」、「orderN.HoppingRange=6.5 Å」、「orderN.KrylovH.order=400」、および RMM-DIISK（混合法)。入力ファイルは「work」ディレクトリの中の「MCCN.dat」です。フーリエ表示での差分電子密度のノルムを SCF ステップの関数として、図 20 に示します。系の電子密度が収束するのに 56 SCF ステップが必要であり、また計算時間は約 7 分でした。この後、以下の様にキーワードを設定します。

```
scf.maxIter                1
scf.EigenvalueSolver    Band
scf.Kgrid              1 1 1
scf.restart               on
MO.fileout                on
num.HOMOs                  2
num.LUMOs                  2
MO.Nkpoint                 1
<MO.kpoint
  0.0  0.0  0.0
MO.kpoint>
```

次に波動関数を計算するために、16 MPI プロセス (Xeon, 2.6GHz) と通常の対角化法を用いて、同じ系の計算を実行し、その計算時間は約 2 分でした。図 21 は上記の方法で計算した MCCN の Γ 点での HOMO および LUMO の等値面図です。この計算例では、O(N) 法と通常の対角化法の計算時間の差はそれほど大きくありませんが、O(N) 法と通常の固有値解法の組み合わせする本方法は数千個以上の原子を含む大規模系に対して有用でしょう。

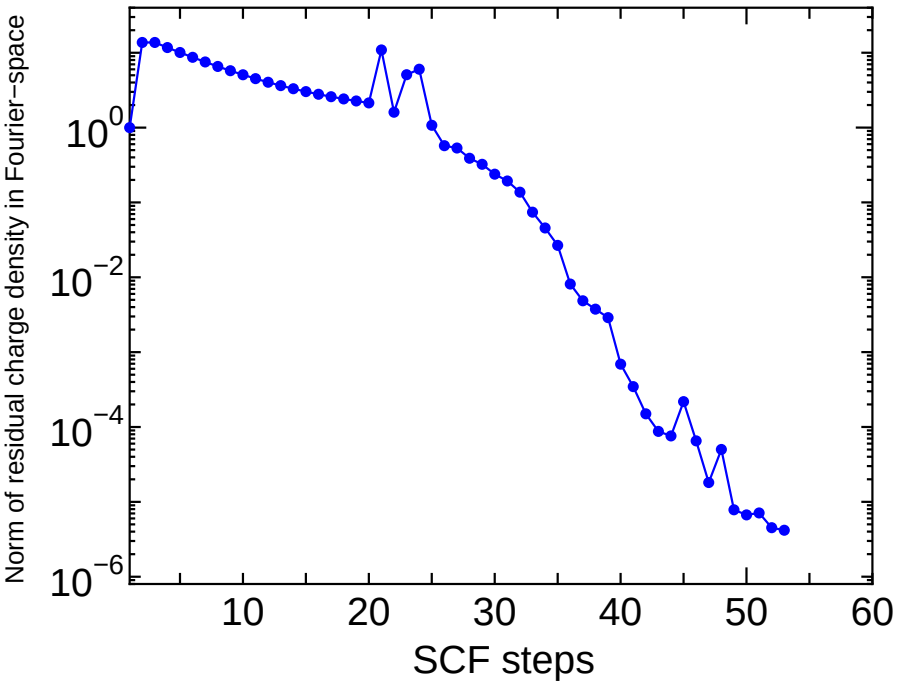


図 20: 564 個の炭素原子で構成される多重連結したカーボンナノチューブ（MCCN）の SCF 収束の過程。フーリエ表示での差分電子密度のノルムを SCF ステップに対して表示。入力ファイルは「work」ディレクトリ中の「MCCN.dat」。

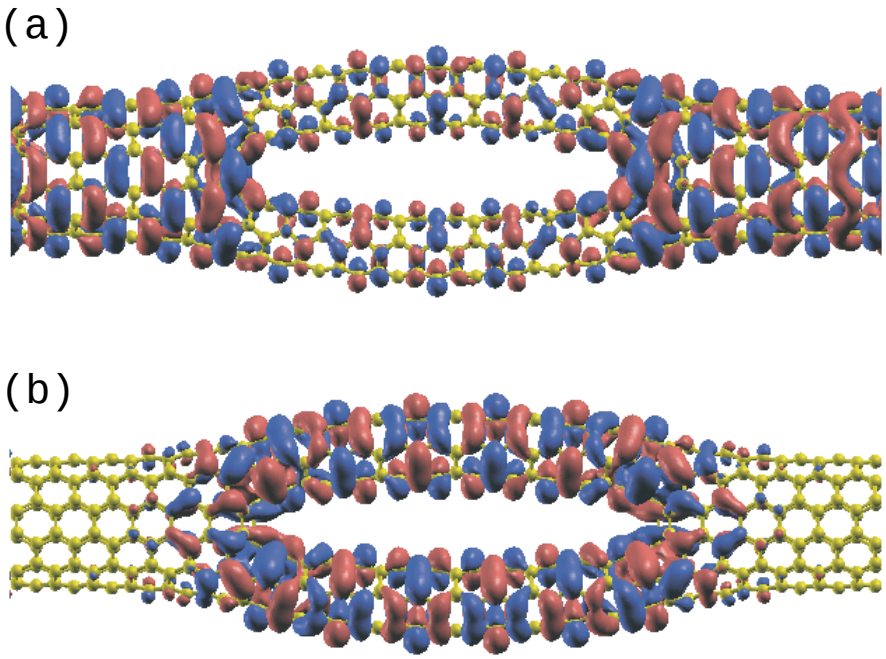


図 21: 564 個の炭素原子で構成される多重連結したカーボンナノチューブ（MCCN）の、(a) 最高被占軌道 (HOMO) および (b) 最低空軌道 (LUMO) の等値面図。等値面値として |0.005| を使用。

24　電場

SCF 計算や構造最適化の際に、ノコギリ波による一様な外部電場を印加することができます。例えば、**a** 軸に沿って 1.0 GV/m （10^9 V/m）の電場を印加する場合、次の様に入力ファイルのキーワード「scf.Electric.Field」を指定して下さい。

```
scf.Electric.Field    1.0 0.0 0.0    # default=0.0 0.0 0.0 (GV/m)
```

電場の符号は、電子に印加されるものとして定義されています。ノコギリ波による一様外部電場が真空領域を持たない周期的なバルク系に印加されると、ポテンシャルの不連続面が導入されることになります。そのため、数値的な不安定性が引き起こされる可能性があります。一方、分子系に対しては不連続面が真空領域に配置されますので、通常は数値不安定性の問題は起こりません。電場印加の例として、電場によって誘起されたニトロベンゼン分子中の全電荷の変化を図 22 に示します。-NO_2 内の酸素とパラ炭素原子、それにパラ水素原子の間で大きな電子移動が起こっているのが分かります。入力ファイルは「work」ディレクトリ中の「Nitro_Benzene.dat」です。図 22 に示した差電子マップに関しては「2 つの Gaussian Cube ファイルの差の解析」も参照して下さい。

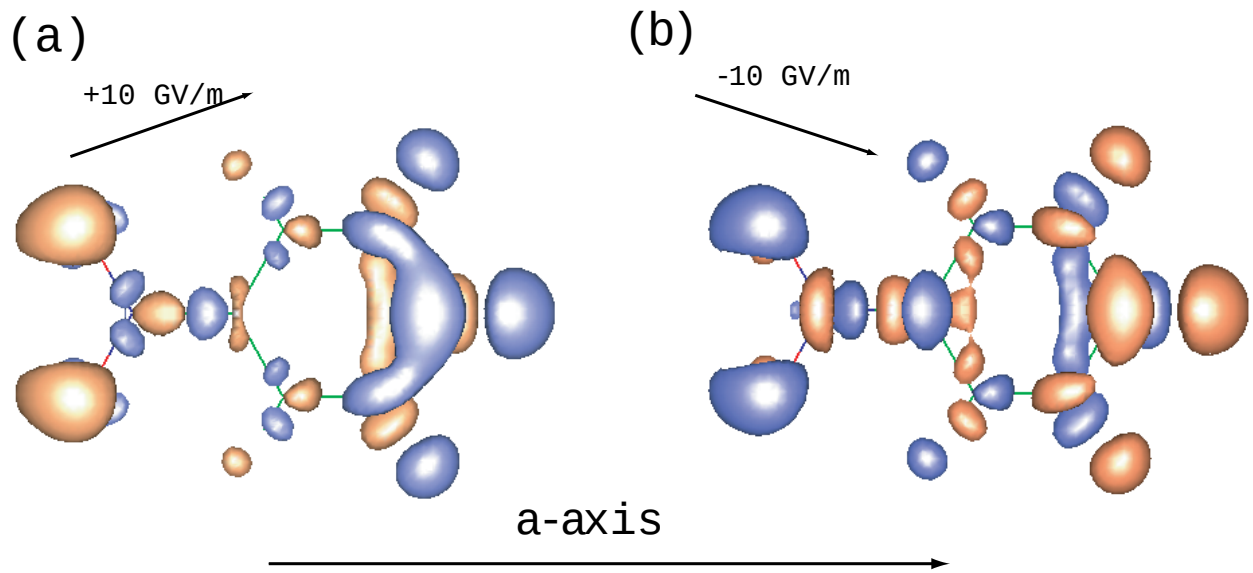


図 22: ゼロ外部電場印加の場合と、**a** 軸に沿って印加された外部電場 (a)10 GV/m、および (b)-10 GV/m 下でのニトロベンゼン分子の全電子密度の差。ここでオレンジ色と青色はそれぞれ、電子密度の増加および減少を示す。傾いた矢印は印加した電場の傾きを図示。入力ファイルは「work」ディレクトリ中の「Nitro_Benzene.dat」。

25　電荷ドーピング

キーワード「scf.system.charge」で電子と正孔のドーピングが可能です。

```
scf.system.charge      1.0      # default=0.0
```

正と負の符号はそれぞれ正孔と電子のドーピングに対応しています。部分的な電荷のドーピングも可能です。OpenMX ではPoisson 方程式の解法にFFT が用いられていますので、キーワード「scf.system.charge」で与えられる過剰電荷は逆電荷を持った一様な背景電荷によって中性化されます。したがって、異なる電荷状態間の全エネルギーを比較する場合には、全エネルギーにはバックグランドの電荷によって誘起される付加的な静電的相互作用が含まれていることに注意が必要です。例として、正孔ドープ、中性、電子ドープの場合のカーボンナノチューブ (14 Å の有限長) のスピン密度を図 23 に示します。中性および電子ドープのナノチューブの全スピンモーメントは 1.0 および 2.2 であり、一方、正孔をドープしたナノチューブでは全スピンモーメントはほとんどゼロです。中性および電子ドープのナノチューブではナノチューブ端のダングリングボンドによりスピン分極が発生しています。

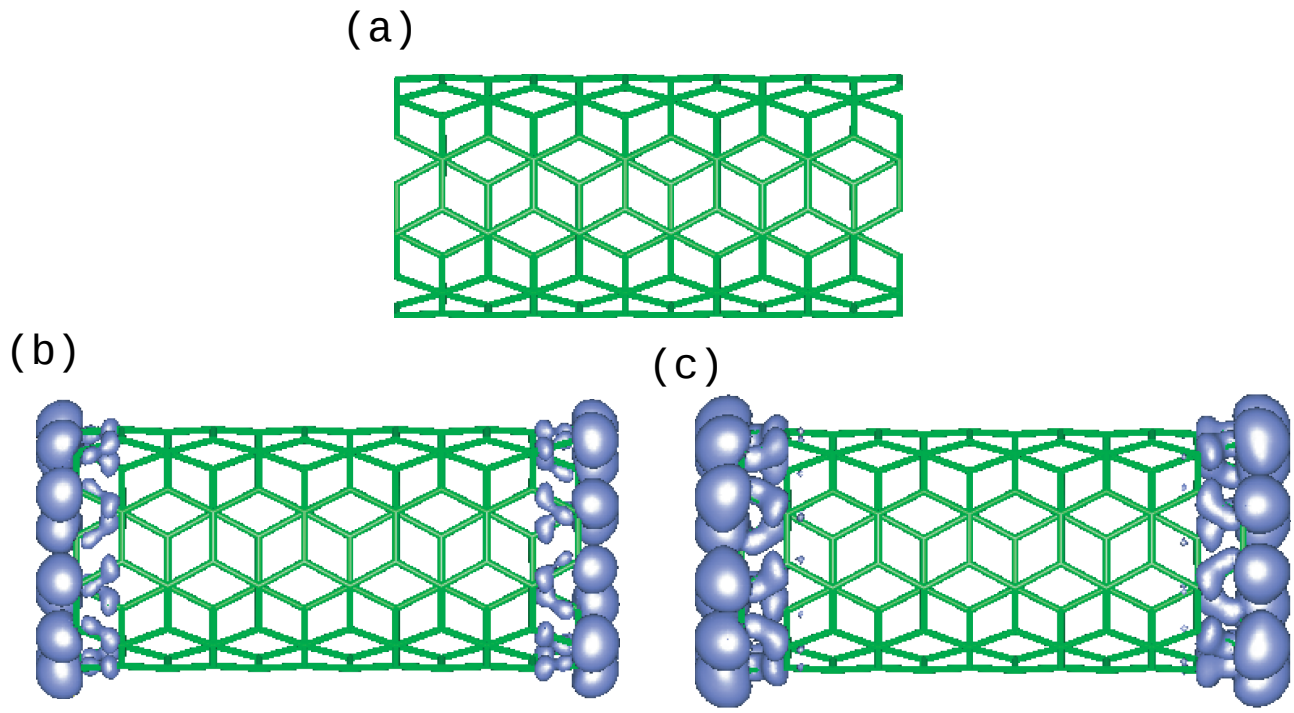


図 23: (a) 4つの正孔をドープした場合、(b) 中性の場合、そして (c) 4つの電子をドープした場合の、14 Å の有限長の (5,5) カーボンナノチューブのスピン密度。インプットファイルは「work」ディレクトリ中の「Doped_NT.dat」。

26　分数核電荷を持つ仮想原子

分数核電荷を持つ仮想原子に対する擬ポテンシャルを使用することで分数核電荷を持つ仮想原子を取り扱うことができます。仮想原子の擬ポテンシャルは ADPACK で生成できます。ADPACK での関連するキーワードは次のように与えられます。

```
AtomSpecies             6.2
total.electron          6.2
valence.electron        4.2
<occupied.electrons
 1    2.0
 2    2.0   2.2
occupied.electrons>
```

上の例は炭素原子と窒素原子の中間に位置付けられる仮想原子に対するものです。仮想原子の擬ポテンシャルに対応する基底関数は同じ分数核電荷を持つ仮想原子に対して生成しなければなりません。これは、*.pao に保存されている原子の電子密度を用いて中性原子のポテンシャルが作成されるためです。例として、この方法をより計算した $C_{7.8}N_{0.2}$ の DOS を図 24 に示します。入力ファイルはディレクトリ「work」中の「DIA8-VA.dat」です。この計算では、単位セル内の 8 個の炭素原子の内の 1 つを、有効核電荷 4.2 の仮想原子に置き換えました。これは $C_{7.8}N_{0.2}$ の化学量論的化合物に相当しています。

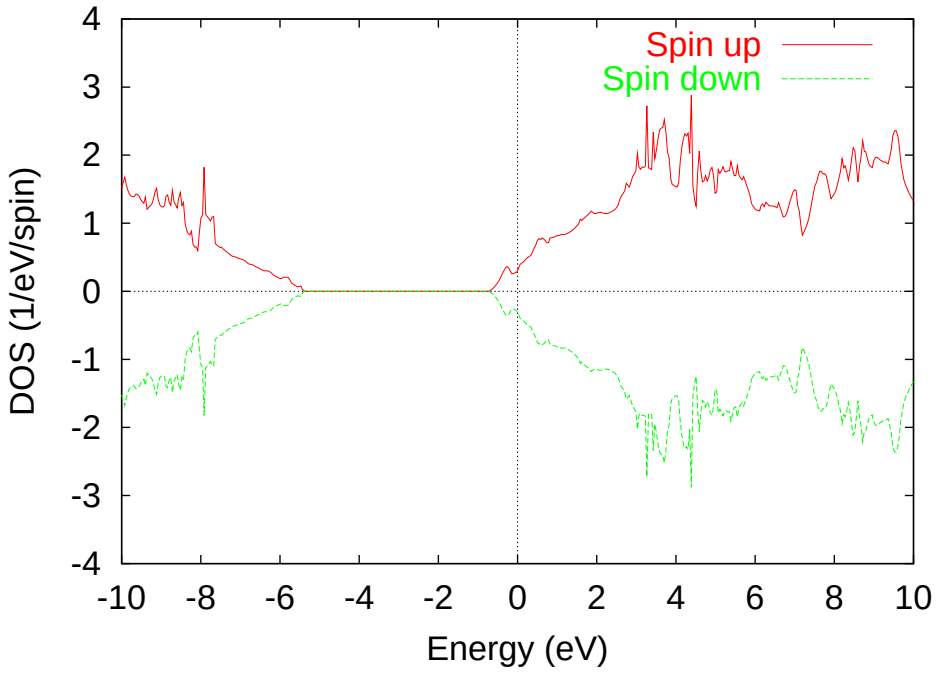


図 24: 仮想原子の擬ポテンシャルを使って計算した $C_{7.8}N_{0.2}$ の状態密度 (DOS)。用いた入力ファイルは「work」ディレクトリ中の「DIA8-VA.dat」。

27　LCAO 係数

クラスター計算とバンド計算の両方で、LCAO 係数を解析することができます。クラスター計算では、キーワード「level.of.fileout」を 2 に設定すると、ファイル「*.out」に LCAO 係数が加えられます。例として、「テスト計算」の章で扱っている「Methane.dat」の LCAO 係数を以下に示します。

```
************************************************************
************************************************************
  Eigenvalues (Hartree) and Eigenvectors for SCF KS-eq.
************************************************************
************************************************************

   Chemical Potential (Hartree) =   0.00000000000000
   HOMO =  4

   LCAO coefficients for up (U) and down (D) spins


                          1 (U)     2 (U)     3 (U)     4 (U)     5 (U)     6 (U)
                        -0.69899  -0.41525  -0.41525  -0.41524   0.21215   0.21215

   1   C 0 s             0.69137  -0.00000   0.00000   0.00000   0.00000   0.00000
         0 px            0.00000  -0.10055   0.63544   0.00033  -0.68649  -1.00467
         0 py            0.00000   0.00028  -0.00029   0.64331   0.00000  -0.00001
         0 pz           -0.00000   0.63544   0.10055  -0.00023  -1.00467   0.68649
   2   H 0 s             0.12870   0.05604  -0.35474  -0.25425  -0.59781  -0.87489
   3   H 0 s             0.12870  -0.35475  -0.05627   0.25420  -0.87488   0.59781
   4   H 0 s             0.12870   0.35497   0.05604   0.25393   0.87488  -0.59781
   5   H 0 s             0.12870  -0.05626   0.35497  -0.25388   0.59781   0.87488

                          7 (U)     8 (U)
                        0.21223   0.24739

   1   C 0 s             0.00000   1.90847
         0 px            0.00000   0.00000
         0 py           -1.21683  -0.00000
         0 pz           -0.00000   0.00000
   2   H 0 s            -0.74926  -0.76083
   ......
   ....
```

バルクの計算で、キーワード「MO.fileout」が ON に設定されている場合、キーワード「MO.kpoint」で指定された **k** 点における LCAO 係数がファイル「*.out」に出力されます。クラスター計算で LCAO 係数を出力するためには「level.of.filout」は 2 でなければなりません。しかし、バンド計算では、関連するキーワードは、「level.of.fileout」ではなく「MO.fikeout」となります。

28　電荷解析

それぞれの原子に有効電荷を割り当てるのは恣意性を含んだ難しい問題ですが、OpenMX においては各原子の電荷状態を解析するために Mulliken 電荷解析、Voronoi 電荷解析、また静電ポテンシャルフィッティング法 (ESP) の 3 つの方法が利用可能です。

28.1　Mulliken 電荷

Mulliken 電荷は、「テスト計算」の章で示したようにデフォルトで「*.out」に出力されます。それぞれの原子に対して射影された Mulliken 電荷に加えて、「*.out」でそれぞれの電子軌道に対して分解された Mulliken 電荷も出力されています。メタン分子の出力ファイル「met.out」に保存された結果を以下に示します。

```
Decomposed Mulliken populations

  1    C          Up spin      Down spin     Sum           Diff
          multiple
s           0    0.598003833  0.598003833   1.196007667   0.000000000
 sum over m      0.598003833  0.598003833   1.196007667   0.000000000
 sum over m+mul  0.598003833  0.598003833   1.196007667   0.000000000
px          0    0.588514081  0.588514081   1.177028163   0.000000000
py          0    0.588703212  0.588703212   1.177406424   0.000000000
pz          0    0.588514081  0.588514081   1.177028162   0.000000000
 sum over m      1.765731375  1.765731375   3.531462749   0.000000000
 sum over m+mul  1.765731375  1.765731375   3.531462749   0.000000000

  2    H          Up spin      Down spin     Sum           Diff
          multiple
s           0    0.409066346  0.409066346   0.818132693   0.000000000
 sum over m      0.409066346  0.409066346   0.818132693   0.000000000
 sum over m+mul  0.409066346  0.409066346   0.818132693   0.000000000

  3    H          Up spin      Down spin     Sum           Diff
          multiple
s           0    0.409065912  0.409065912   0.818131824   0.000000000
 sum over m      0.409065912  0.409065912   0.818131824   0.000000000
 sum over m+mul  0.409065912  0.409065912   0.818131824   0.000000000

 .......
 ....
```

ここで分かるように、Mulliken 電荷は全ての基底関数に対して分解されています。上記の表では分解された Mulliken 電荷は 2 種類の方法で加算されています。その一つは「sum over m」で、それぞれの動径関数に対する磁気量子数についての加算を意味しています。もう一つは「sum over m+mul」で、これは磁気量子数と動径関数についての加算を意味しています。

28.2 Voronoi 電荷

各原子の Voronoi 電荷は、Voronoi 多面体内の電子密度を積分することで計算されます。OpenMX での Voronoi 多面体は Fuzzy セル分割法 [49] による**連続的にぼかされた** (smeared) 境界から構成されています。Voronoi 多面体は原子半径を考慮せずに構造だけで決定されるので、Voronoi 解析はしばしば過大あるいは過小評価した電荷を与えることに注意する必要があります。Voronoi 電荷を計算する場合には、入力ファイルに次のキーワード「Voronoi.charge」を設定してください。

```
Voronoi.charge      on       # on|off, default = off
```

メタン分子の場合、以下の Voronio 電荷が「*.out」に出力されます。

```
***********************************************************
***********************************************************
                    Voronoi charges
***********************************************************
***********************************************************

  Sum of Voronoi charges for up    =  3.999999031463
  Sum of Voronoi charges for down  =  3.999999031463
  Sum of Voronoi charges for total =  7.999998062926

  Total spin S by Voronoi charges  =  0.000000000000

                    Up spin      Down spin     Sum          Diff
      Atom=   1   1.137912511  1.137912511   2.275825021  0.000000000
      Atom=   2   0.715521700  0.715521700   1.431043399  0.000000000
      Atom=   3   0.715521486  0.715521486   1.431042973  0.000000000
      Atom=   4   0.715521776  0.715521776   1.431043552  0.000000000
      Atom=   5   0.715521559  0.715521559   1.431043118  0.000000000
```

明らかに炭素原子（Atom = 1）は小さい電荷を、水素原子（Atom=2-5）は大きい電荷を持っており、この結果は化学的な直観とは整合しません。しかし、Voronoi 解析は最密構造を持つバルク固体に対して、補足的な情報となるでしょう。

28.3 静電ポテンシャルフィッティング

小さな分子系に対しては、静電ポテンシャル (ESP) フィッティング法 [65, 66, 67] が各原子の有効電荷を決定するのに良く用いられています。多くの場合に、得られた結果は化学的な直観に整合しています。ただし、ESP フィッティング法では表面部分から離れた原子に対して、十分なサンプリング点が存在しないために、この方法は巨大分子やバルク系には適用できません。ESP フィッティング法では、有効点電荷による静電ポテンシャルの和が、可能な限り DFT 計算によって求めた静電ポテンシャルを再現するように、各原子の正味の有効点電荷を最小二乗法で決定します。ESP フィッティング電荷は次の二段階で計算します。

(1) SCF 計算

通常の SCF 計算後、次の 2 つのファイルが出力されます。

```
*.out
*.vhart.cube
```

二つのファイルを作成するための追加のキーワードは必要ありません。これらのファイルは通常の SCF 計算から得られるの出力ファイルですが、キーワード「level.of.std.out」は 1 または 2 に設定する必要があります。

(2) ESP フィッティング電荷

ESP フィッティング電荷を計算するためにプログラムコードをコンパイルします。ディレクトリ「source」に移動し、次のようにコンパイルします。

```
% make esp
```

コンパイルが正常に完了すると、ディレクトリ「work」に実行形式ファイル「esp」が作成されます。ESP フィッティング電荷はプログラム「esp」を使って、二つのファイル「*.out」、「*.vhart.cube」から計算します。例えば、「テスト計算」の章で示したメタン分子の場合、ESP フィッティング電荷は次の様に計算されます。

```
% ./esp met -c 0 -s 1.4 2.0
```

拡張子なしのファイル名を指定するだけで十分ですが、二つのファイル「met.out」と「met.vhart.cube」は「work」ディレクトリに保存しておかなければなりません。オプションの「-c」と「-s」は制限条件と倍率 (scale factor) を指定するパラメータです。ソースコード「eps.c」のヘッダーの部分に次の説明が記載されています。

```
-c      constraint parameter
        '-c 0' means charge conservation
        '-c 1' means charge and dipole moment conservation
-s      scale factors for vdw radius
```

```
'-s 1.4 2.0' means that 1.4 and 2.0 are 1st and 2nd scale factors
```

ESP フィッティング法は、電荷保存、並びに電荷と双極子モーメント保存の二つの制約条件下でフィッティングを行います。後者は DFT 計算によって求めた電荷と双極子モーメントを再現できますが、双極子モーメントの保存条件を導入すると、特に大きな分子に対して、物理的に容認できない点電荷が得られる可能性が生じます。したがって前者の制約条件が推奨されます。サンプリング点としては二つのシェル構造で挟まれた領域内の実空間格子点が選択されます。シェル構造は各原子に割り当てられた球の足し合せから定義されます。球の半径は「-s」以降に与えた 1 番目および 2 番目の倍率 (scale factor)× ファン・デル・ワールス半径で与えられます [68]。

計算結果は次のように標準出力（ユーザーのディスプレイ）に出力されます。

```
 % ./esp met -c 0 -s 1.4 2.0

**********************************************************************
**********************************************************************
 esp: effective charges by a ESP fitting method
 Copyright (C), 2004, Taisuke Ozaki
 This is free software, and you are welcome to
 redistribute it under the constitution of the GNU-GPL.
**********************************************************************
**********************************************************************

Constraint: charge
Scale factors for vdw radius    1.40000    2.00000
Number of grids in a van der Waals shell = 28464
Volume per grid =    0.0235870615 (Bohr^3)
Success

  Atom=   1  Fitting Effective Charge= -0.93558216739
  Atom=   2  Fitting Effective Charge=  0.23389552572
  Atom=   3  Fitting Effective Charge=  0.23389569182
  Atom=   4  Fitting Effective Charge=  0.23389535126
  Atom=   5  Fitting Effective Charge=  0.23389559858

  Magnitude of dipole moment    0.0000015089 (Debye)
  Component x y z     0.0000003114   -0.0000002455   -0.0000014558
RMS between the given ESP and fitting charges (Hartree/Bohr^3)= 0.096515449505
```

29 ノンコリニア DFT

OpenMX ではノンコリニア DFT 法が利用可能です。各実空間グリッド上での量子化軸に制限を持たない最も一般的な方法が実装されています。またスピン軌道相互作用 [6, 7, 8, 9, 13] も自己無撞着計算において取り入れることが可能です。ノンコリニア DFT 計算を実行する場合、キーワード「scf.SpinPolarization」を以下のように設定して下さい。

```
scf.SpinPolarization        NC        # On|Off|NC
```

この場合、Kohn-Sham 軌道は二成分スピノルによって表現されます。それぞれのサイトの初期スピンの方位は次のように与えられます。

```
<Atoms.SpeciesAndCoordinates           # Unit=Ang
  1  Mn    0.00000   0.00000   0.00000   8.0  5.0  45.0 0.0 45.0 0.0  1 on
  2  O     1.70000   0.00000   0.00000   3.0  3.0  45.0 0.0 45.0 0.0  1 on
Atoms.SpeciesAndCoordinates>
```

1: 原子の通し番号
2: 原子種の名前
3: x-座標
4: y-座標
5: z-座標
6: up スピン状態に対する初期占有数
7: down スピン状態に対する初期占有数
8: スピン磁気モーメントに対する局所磁場の Euler 角: theta
9: スピン磁気モーメントに対する局所磁場の Euler 角: phi
8 と 9 番目の Euler 角は初期スピン密度の生成にも使用
10: 軌道磁気モーメントに対する局所磁場の Euler 角: theta
11: 軌道磁気モーメントに対する局所磁場の Euler 角: phi
12: 次のいずれかのキーワード「scf.Constraint.NC.Spin」、「scf.NC.Zeeman.Orbital」、もしくは「scf.NC.Zeeman.Orbital」が指定された際に、それぞれの原子に対して制約法を適用する場合には「1」、適用しない場合には「0」。
13: LDA(GGA)+U 法の際に、軌道分極を促進する場合には「on」、通常計算の場合には「off」。

スピン磁気モーメントと軌道磁気モーメントの方位に対する初期 Euler 角 (θ、ϕ) はそれぞれ 8 番目、9 番目および 10 番目、11 番目の列で与えられます。12 番目の列は制約法のスイッチで、それぞれの原子サイトにスピン磁気モーメントと軌道磁気モーメントの方位に対する制約汎関数を付加します。ここで、「1」は制約汎関数を付加、「0」は制約法無しを意味します。スピン磁気モーメントの方位に制約条件を付加する DFT 計算の詳細については「スピン磁気モーメント方位の制約 DFT」のセクションを参照し

て下さい。最後の 13 番目の列は LDA(GGA)+U 法の際に、軌道分極を促進する手法に対するスイッチで、軌道分極を促進する場合には「on」、通常計算の場合には「off」となります。図 25 はノンコリニア DFT 法で計算した MnO 分子中のスピン磁気モーメントをベクトル表示したものです。この計算は、「work」ディレクトリ内の入力ファイル「Mol_MnO_NC.dat」を使って、再現することができます。実空間でノンコリニアな方位を持つスピン磁気モーメントを可視化するために、次の二つのファイルが利用できます。

```
*.nc.xsf
*.ncsden.xsf
```

ここで、*はユーザーが指定した「System.Name」です。「*.nc.xsf」と「*.ncsden.xsf」は XCrySDen でサポートされているベクトルファイルフォーマット (XSF) 形式のファイルです。前者が Mulliken 解析によってそれぞれの原子に対して射影されたスピン磁気モーメントを、後者が実空間グリッド上のスピン磁気モーメントになります。この二つのファイルは、図 25 に示すように XCrySDen の「Display → Forces」を使って可視化できます。

Mulliken 解析で計算されるそれぞれの原子のスピン磁気モーメントとその Euler 角は、ファイル「*.out」内に次のように保存されています。

```
************************************************************
************************************************************
                  Mulliken populations
************************************************************
************************************************************

  Total spin moment (muB)   4.998503442   Angles (Deg) 44.991211196   0.000000000

              Up       Down      Sum      Diff        theta      phi
   1   Mn   9.59803  4.76902  14.36705   4.82901    44.99208    0.00000
   2    O   3.40122  3.23173   6.63295   0.16949    44.96650   -0.00000
```

一般にノンコリニア DFT 法おける SCF 計算はコリニア計算と比較して、収束が容易ではありません。これはスピン方位の変化に伴うエネルギーの変化が非常に小さいためです。スピン磁気モーメントや軌道磁気モーメントの方位に制約条件を課す制約法は SCF 計算の収束の加速に有効に働くことがあります。　ノンコリニア DFT 法では、スピン軌道相互作用がサポートされていますが、コリニア DFT 法ではサポートされていません。スピン軌道相互作用に関しては「相対論的効果」の章を参照して下さい。

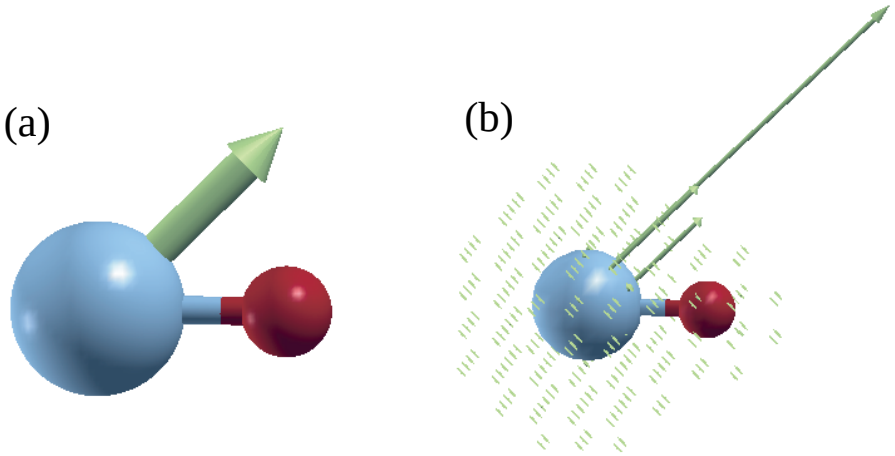


図 25: ノンコリニア DFT 法で計算した MnO 分子中のスピン磁気モーメント。(a) Mulliken 解析によってそれぞれの原子に対して射影されたスピン磁気モーメント。(b) 実空間グリッド上のスピン磁気モーメント。図は XCrySDen の「Display → Forces」により可視化。入力ファイルは「work」ディレクトリ内の「Mol_MnO_NC.dat」。

30 相対論的効果

OpenMX の web サイトで公開されている擬ポテンシャルは原子に対する Dirac 方程式に基づき生成されています。相対論的擬ポテンシャルには質量項やポテンシャル勾配項から生じる相対論効果に加え、スピン軌道相互作用も含まれています。この相対論的擬ポテンシャルを用いて相対論効果を考慮することが可能です。ただし OpenMX の計算では波動方程式として Shrödinger 方程式と同等の微分演算子が用いられていますので、価電子に対する直接的な相対論効果は無視されています。直接的な効果は小さいことが分かっており、価電子に対する大部分の相対論効果は内殻電子を通し、間接的に生じます。間接的な相対論効果は擬ポテンシャルの中に含まれており、結局、内殻電子だけでなく OpenMX の計算で扱う価電子に対してもまた十分な精度で相対論効果が考慮されています。スピン軌道相互作用を量子数 j の縮退度に応じて平均化する方法は「半相対論的扱い」として知られ、またスピン軌道相互作用を含めた場合には「完全な相対論的扱い」と呼ばれます。コリニア DFT 計算の際には「半相対論的扱い」のみが実行できます。またノンコリニア DFT 計算の際には「半相対論的扱い」と「完全な相対論的扱い」のどちらも利用可能です。

30.1 完全な相対論的扱い

擬ポテンシャル法の範疇で、スピン軌道相互作用を含む「完全な相対論効果」をノンコリニア DFT 計算において考慮することが出来ます。 [10, 19, 13]。一方、スピン軌道相互作用の取扱いはコリニア DFT 計算ではサポートされていません。次の二つのステップで「完全な相対論効果」の考慮が可能です。

(1) j 依存型擬ポテンシャルの生成

まず最初に ADPACK を使用し、j 依存型擬ポテンシャルを作成して下さい。ユーザーの便宜のために、多くの元素に対する j 依存型擬ポテンシャルがデータベース Ver. 2013 [89] として利用可能です。j 依存型擬ポテンシャルを作成するための詳細は ADPACK のマニュアルを参照して下さい。

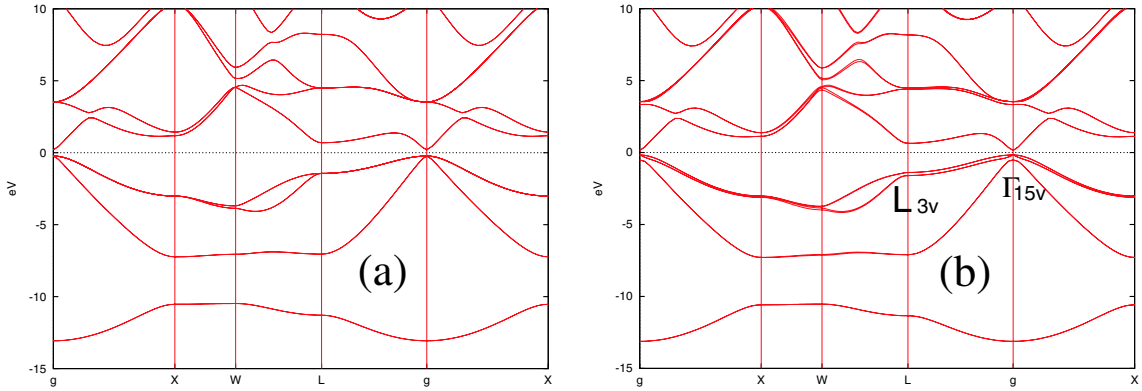


図 26: ノンコリニア DFT 法によって計算したスピン軌道相互作用が (a) 無い場合と、(b) ある場合の GaAs 固体のバンド構造。Ver.2013 のデータベースから、基底関数として Ga7.0-s2p2d2 と As7.0-s2p2d2 を、また擬ポテンシャルとして Ga_CA13.vps と As_CA13.vps を使用。交換相関項は LDA を、scf.Kgrid と scf.energycutoff にはそれぞれ、12× 12× 12 と 140 (Ryd) を使用。また計算における格子定数は実験値 (5.65Å)。入力ファイルは「work」ディレクトリの「GaAs.dat」。

表 3: GaAs 固体の Γ_{15v} および L_{3v} において計算されたスピン軌道分裂 (eV) 。比較のために他の理論値 (LMTO: 参考文献 [69], PP: 参考文献 [70]) と実験値 (参考文献 [71]) も併記。入力ファイルは「work」ディレクトリ中の「GaAs.dat」。

Level	OpenMX	LMTO	PP	Expt.
Γ_{15v}	0.344	0.351	0.35	0.34
L_{3v}	0.213	0.213	0.22	

(2) SCF 計算

キーワード「Definition.of.Atomic.Species」の設定中で j 依存型擬ポテンシャルを指定した場合、次のキーワード「scf.SpinOrbit.Coupling」でスピン軌道相互作用が導入できます。

```
scf.SpinOrbit.Coupling      on          # On|Off, default=off
```

スピン軌道相互作用は、摂動法として取り扱われるのではなく、擬ポテンシャル法の枠組で自己無撞着に組み込まれます。このスピン軌道相互作用を介して、二成分スピノル中の α および β スピン成分は直接に相互作用します。ノンコリニアＤＦＴ計算において実空間でのスピン方位はスピン軌道相互作用を組み込んだ場合に意味を持ちます。スピン軌道相互作用が存在しない場合には、全てのスピン方位はエネルギー的に縮退しており、実空間でスピン方位は一義的に決定されません。スピン軌道分裂の例として、図 26 にスピン軌道相互作用がある場合と無い場合のノンコリニア DFT 法によって計算した GaAs 固体のバンド構造を示します。入力ファイルは「work」ディレクトリの「GaAs.dat」です。図 26(a) ではスピン軌道相互作用による分裂は見られませんが、図 26(b) ではいくつかのバンドの縮退が解けていることが見てとれます。表 3 に二つの $\mathbf{k}$ 点、Γ 点および L 点でのスピン軌道相互作用によるエネルギー分裂を示します。他の計算値および実験値と良い一致が見て取れます。

30.2　半相対論的扱い

「半相対論的な取り扱い」を導入する方法の一つは、ADPACK によって作成された半相対論的擬ポテンシャルを使用することですが、より簡単な方法は j 依存型擬ポテンシャルを用い、次のようにキーワード「scf.SpinOrbit.Coupling」を「off」にすることです。

```
scf.SpinOrbit.Coupling      off         # On|Off, default=off
```

この場合、OpenMX によって j 依存型擬ポテンシャルが読み込まれた際に、自動的に j の縮退の重みで平均化されます。この平均化の操作によって j 依存型擬ポテンシャルは半相対論的な擬ポテンシャルに変換されます。つまり、j 依存型擬ポテンシャルは、相対論的および半相対論的取り扱いの両者に利用できることになります。したがって、相対論的な効果を考慮する場合には、半相対論的な擬ポテンシャルではなく、完全に相対論的な j 依存型擬ポテンシャルを作成することをお勧めします。実際、図 26(a) の計算は「scf.SpinOrbit.Coupling=off」とし、図 26(b) と同じ擬ポテンシャルのファイルを用いて、実行したものです。

31　軌道磁気モーメント

各原子サイトでの軌道磁気モーメントは、ノンコリニア DFT 計算において、デフォルトで計算されます。局所的な軌道磁気モーメントはスピン軌道相互作用の結果として現れるため、スピン軌道相互作用が考慮された場合にのみ、この値は有限値となります [74, 75]。例として、ディレクトリ「work」内の入力ファイル「FeO_NC.dat」を用いた酸化鉄固体のノンコリニア LDA+U 計算 (U=5eV) について説明します。LDA+U の計算については、「LDA+U」の章を参照して下さい。計算した鉄のサイトの軌道磁気モーメントおよびスピン磁気モーメントを表 4 に示します。また以下のように「*.out」内に軌道成分に分解した軌道磁気モーメントが出力されています。ここで「*」は「System.Name」を意味します。

```
***********************************************************
***********************************************************
                     Orbital moments
***********************************************************
***********************************************************

  Total Orbital Moment (muB)   0.000001885   Angles  (Deg) 126.954120326  185.681623854

        Orbital moment (muB)   theta (Deg)  phi (Deg)
   1   Fe   0.76440           131.30039    51.57082
   2   Fe   0.76440            48.69972   231.57071
   3    O   0.00000            40.68612   210.48405
   4    O   0.00000            48.18387   222.72367


  Decomposed Orbital Moments

    1   Fe          Orbital Moment(muB)    Angles (Deg)
           multiple
  s           0    0.000000000           90.0000    0.0000
   sum over m      0.000000000           90.0000    0.0000
  s           1    0.000000000           90.0000    0.0000
   sum over m      0.000000000           90.0000    0.0000
  px          0    0.000055764           42.7669  270.0000
  py          0    0.000046795           28.9750  180.0000
  pz          0    0.000044132           90.0000  239.0920
   sum over m      0.000120390           47.1503  239.0920
  px          1    0.001838092           10.8128  -90.0000
  py          1    0.001809013            3.5933  180.0000
  pz          1    0.000362989           90.0000  251.7994
```

表 4: 遷移金属酸化物 MO (M=Mn, Fe, Co, Ni) のスピン磁気モーメント $M_s(\mu_B)$ と軌道磁気モーメント $M_o(\mu_B)$。LDA+U 法 [16] を用いて計算し、M の第 1d 軌道に対して、 Mn には 3.0 (eV) 、Fe には 5.0 (eV)、Co には 7.0 (eV)、Ni には 7.0 (eV) の実効 U 値を使用。Mulliken 解析は多重基底関数の使用に際して大きめのスピンモーメントを与える傾向があるため、Voronoi 分割によって、局在スピンモーメントを計算。入力ファイルは「work」ディレクトリ中の「MnO_NC.dat」、「FeO_NC.dat」、「CoO_NC.dat」、「NiO_NC.dat」。比較のために他の理論値 [50] 実験値 [50] も併記。

	M_s		M_o		
Compound	OpenMX	Other calc.	OpenMX	Other calc.	Expt. in total
MnO	4.519	4.49	0.004	0.00	4.79,4.58
FeO	3.653	3.54	0.764	1.01	3.32
CoO	2.714	2.53	1.269	1.19	3.35,3.8
NiO	1.687	1.53	0.247	0.27	1.77,1.64,1.90

```
 sum over m      0.003683170          11.3678  251.7994
d3z^2-r^2   0    0.043435663          90.0000  224.2874
dx^2-y^2    0    0.066105902          24.3591  229.7056
dxy         0    0.361874370          80.4206   50.6465
dxz         0    0.397108491         144.2572  -12.7324
dyz         0    0.427070801         138.9995  100.0151
 sum over m      0.776513038         132.4577   51.6984
d3z^2-r^2   1    0.000144144          90.0000  196.4795
dx^2-y^2    1    0.000270422          31.2673  224.0799
dxy         1    0.003006770          85.5910   50.2117
dxz         1    0.002952926         139.3539   -4.1301
dyz         1    0.003222374         134.0513   95.9246
 sum over m      0.006795789         126.2536   52.1993
.....
...
```

表 4 に示すように、OpenMX の計算結果は、一連の $3d$ 遷移金属酸化物のスピンおよび軌道磁気モーメントの他の計算結果と良く一致しています。しかし、軌道磁気モーメントの絶対値は基底関数や DFT+U 法のオンサイトの「U」などの計算条件に大きく影響を受けます。軌道磁気モーメントの収束計算には分極関数を含むかなり大きな基底関数が必要であることにも注意すべきです。一方、スピン磁気モーメントの値は計算条件に対する変化が比較的、小さいことが経験上、分かっています。

32 LDA+U

LDA(GGA)+U 法 [16] がキーワード「scf.Hubbard.U」によってコリニア計算とノンコリニア計算で実行可能です。

```
scf.Hubbard.U              on       # On|Off, default=off
```

LDA+U 法は LDA だけでなく GGA にも適用可能です。占有数を計算するための占有数演算子の選択は次のキーワード「scf.Hubbard.Occupation」によって指定します。

```
scf.Hubbard.Occupation     dual     # onsite|full|dual, default=dual
```

3 つの占有数演算子中で、双対演算子 (dual) のみが電子数の総和則を満たし、この場合には占有数行列の対角成分の和が全電子数となります。この総和則は Hubbard モデルで最も基本的な保存量です。占有数演算子「onsite」、「full」、「dual」の詳細については文献 [16] を参照して下さい。原子種を次のように定義した場合、

```
<Definition.of.Atomic.Species
 Ni  Ni6.0S-s2p2d2     Ni_CA13S
 O   O5.0-s2p2d1       O_CA13
Definition.of.Atomic.Species>
```

それぞれの軌道に対する実効 U 値 (eV 単位) は、以下のように指定されます。

```
<Hubbard.U.values                  #  eV
 Ni  1s 0.0 2s 0.0 1p 0.0 2p 0.0 1d 4.0 2d 0.0
 O   1s 0.0 2s 0.0 1p 0.0 2p 0.0 1d 0.0
Hubbard.U.values>
```

記述の最初は「<Hubbard.U.values」で、最後は「Hubbard.U.values>」となります。ユーザーは、全ての基底関数に対して、上記のフォーマットで実効 U 値 (eV 単位) を与えなければなりません。「1s」と「2s」は第 1 および第 2 番目の s 軌道を、「1s」に続く数値は第 1 番目の s 軌道に対する実効 U 値です。同じ規則が p および d 軌道にも適用されます。LDA+U 法に基づく計算例として、酸化ニッケル固体の状態密度を図 27 に示します。ここでニッケルの d 軌道に対する実効 U 値として 0 (eV) と 4 （eV）が使用されました。入力ファイルは「work」ディレクトリ中の「Crys-NiO.dat」です。d 軌道に Hubbard 項を導入したことで、ギャップが大きくなっていることが分かります。各原子サイトの占有数行列の固有値と固有ベクトルは、次のようにファイル「*.out」中に、タイトル「Occupation Number in LDA+U and Constraint DFT」から始まり、出力されています。

```
***********************************************************
***********************************************************
       Occupation Number in LDA+U and Constraint DFT
```

```
   Eigenvalues and eigenvectors for a matrix consisting
          of occupation numbers on each site
*************************************************************
*************************************************************

   1   Ni

    spin= 0

 Sum =  8.591857905308

                  1       2       3       4       5       6       7       8
 Individual    -0.0024  0.0026  0.0026  0.0038  0.0051  0.0051  0.0888  0.0950

 s          0   0.1671  0.0005 -0.0006  0.0040  0.0000  0.0005 -0.0124  0.0000
 s          1  -0.9856 -0.0030  0.0039 -0.0227 -0.0000 -0.0072  0.0066  0.0000
 px         0   0.0010  0.0004  0.0011 -0.0131  0.0004  0.0001 -0.0261 -0.0291
 py         0   0.0010  0.0006 -0.0008 -0.0130  0.0000  0.0009 -0.0271 -0.0000
 pz         0   0.0010 -0.0012 -0.0001 -0.0131 -0.0004  0.0001 -0.0261  0.0291
 px         1   0.0067  0.0023  0.0066 -0.0792 -0.0161  0.0123  0.5594  0.7062
 py         1   0.0068  0.0041 -0.0053 -0.0801 -0.0000 -0.0162  0.5797  0.0002
 pz         1   0.0067 -0.0070 -0.0005 -0.0792  0.0161  0.0123  0.5594 -0.7063
 d3z^2-r^2  0   0.0002 -0.0781 -0.0105  0.0002  0.0023  0.0014  0.0002  0.0108
 dx^2-y^2   0   0.0004 -0.0105  0.0781  0.0004 -0.0013  0.0024  0.0003 -0.0062
 dxy        0   0.0004 -0.0009 -0.0002  0.0246 -0.0421 -0.0251  0.0794 -0.0050
 dxz        0  -0.0001  0.0008 -0.0010  0.0269  0.0000  0.0478  0.0795  0.0000
 dyz        0   0.0004  0.0004  0.0008  0.0246  0.0420 -0.0251  0.0794  0.0050
 d3z^2-r^2  1  -0.0023  0.9875  0.1327 -0.0033 -0.0262 -0.0159 -0.0001 -0.0069
 dx^2-y^2   1  -0.0040  0.1326 -0.9875 -0.0056  0.0151 -0.0275 -0.0002  0.0040
 dxy        1   0.0091  0.0233  0.0052 -0.5578  0.7055  0.4249 -0.0749  0.0157
 dxz        1   0.0189 -0.0180  0.0233 -0.5964 -0.0003 -0.7958 -0.0748 -0.0000
 dyz        1   0.0091 -0.0110 -0.0212 -0.5578 -0.7052  0.4255 -0.0749 -0.0157

                  9      10      11      12      13      14      15      16
 Individual     0.0952  0.2456  0.9902  0.9974  0.9975  1.0060  1.0060  1.0137

 s          0   0.0002  0.9859 -0.0036 -0.0001  0.0000 -0.0000  0.0000 -0.0000
 .....
 ...
```

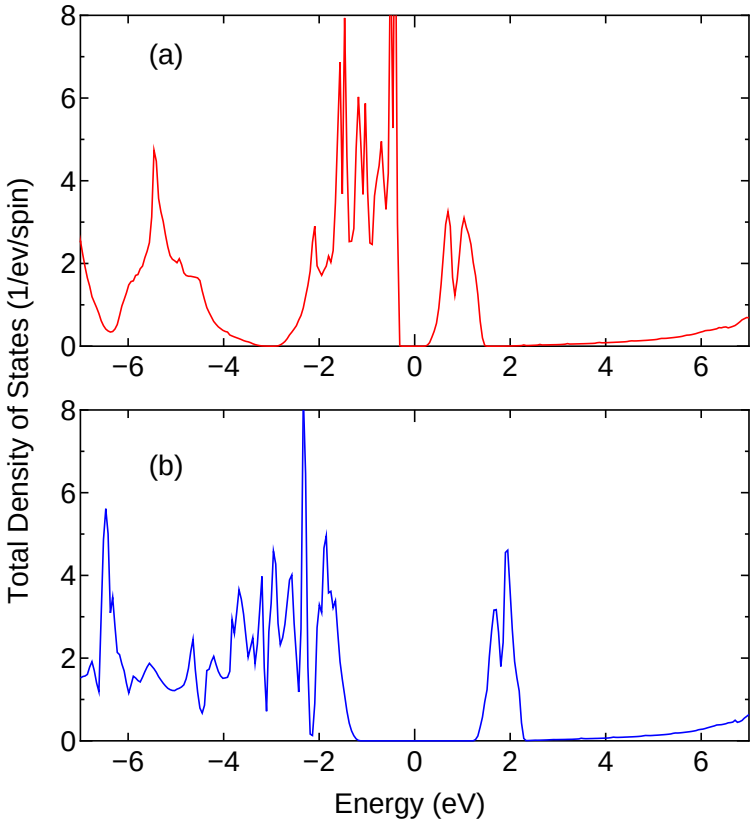


図 27: LDA+U 法を使い、(a) U=0 (eV)、(b) U=4 (eV) で計算された、NiO 固体のアップスピンの全状態密度。入力ファイルは「work」ディレクトリ中の「Crys-NiO.dat」。

それぞれの原子サイトの占有数行列の固有値は、対応する固有ベクトルによって与えられた局所状態に対する占有数となります。LDA+U 法における汎関数は、軌道占有の自由度に複数の極値を持っており、SCF 計算の結果は初期占有数近傍に存在するある極値に収束することがあります。そのため収束した電子状態は最低エネルギーを持った基底状態であるとは限りません。軌道分極を持つ基底状態を見出すために、次のスイッチによって、軌道分極を明示的に促進する方法が利用できます。

コリニア計算の場合

```
<Atoms.SpeciesAndCoordinates            # Unit=AU
 1  Ni    0.0        0.0        0.0       10.0  6.0  on
 2  Ni    3.94955    3.94955    0.0        6.0 10.0  on
 3   O    3.94955    0.0        0.0        3.0  3.0  on
 4   O    3.94955    3.94955    3.94955    3.0  3.0  on
Atoms.SpeciesAndCoordinates>
```

ノンコリニア計算の場合

```
<Atoms.SpeciesAndCoordinates           # Unit=AU
 1  Ni   0.0       0.0       0.0       10.0  6.0  40.0 10.0 0 on
 2  Ni   3.94955   3.94955   0.0        6.0 10.0  40.0 10.0 0 on
 3   O   3.94955   0.0       0.0        3.0  3.0  10.0 40.0 0 on
 4   O   3.94955   3.94955   3.94955    3.0  3.0  10.0 40.0 0 on
Atoms.SpeciesAndCoordinates>
```

軌道分極を促進させる場合には、最後の列で「on」と指定して下さい。通常の計算を行う場合には「off」と指定します。軌道分極の促進は各原子サイト毎に行われるため、全ての原子に対して「on」もしくは「off」を指定しなければなりません。指定を明示的に行わない場合には、「off」が設定されます。「on」の設定がされた場合は、最初の数回の SCF ステップにおいて軌道分極が促進され、それ以降に続く SCF ステップでは通常の計算が行われます。この取扱いにより軌道分極が促進され、多くの場合で基底状態が得られますが、逆に不安定な方向への収束を導く可能性もあり、万能な方法ではありません。詳細は文献 [16] を参照して下さい。

33　ノンコリニアスピン方位に対する制約条件付き DFT

ノンコリニア DFT 法において任意のスピン方位を持つ電子構造を計算するために、OpenMX Ver. 3.7 では制約条件付き DFT 法が利用可能です。制約条件付き DFT 法では、スピン方位と初期スピン方位の差がゼロで無い限り系のエネルギーにペナルティーを与える汎関数が付加されます [11]。ノンコリニアスピン方位に対する制約条件付き DFT 法は次のキーワードで利用できます。

```
scf.Constraint.NC.Spin        on      # on|off, default=off
scf.Constraint.NC.Spin.v    0.5       # default=0.0(eV)
```

キーワード「scf.Constraint.NC.Spin」のスイッチをオンにし、また制約の強さを決定するキーワード「scf.Constraint.NC.Spin.v」によってその大きさを指定すれば、スピン方位に対する制約条件付き汎関数が付加されます。制約条件は次のようにスイッチを指定する事で、各原子に適用されます。

```
<Atoms.SpeciesAndCoordinates
   1    Cr     0.00000    0.00000    0.00000  7.0  5.0 -20.0 0.0  1  off
   2    Cr     0.00000    2.00000    0.00000  7.0  5.0  20.0 0.0  1  off
Atoms.SpeciesAndCoordinates>
```

第 10 列の「1」は制約有り、「0」は無しを意味しています。この方法はスピンの方位だけに制約条件を導入し、スピンの大きさには制約を課しません。また、この制約法は「LDA+U」の章で説明した LDA+U 計算と同時に適用することが可能です。実例として、2 つの局所スピン間の相対角に対するクロム 2 量体の全エネルギーと磁気モーメントの依存性を図 28 に示します。この計算は「work」ディレクトリ中の入力ファイル「Cr2_CNC.dat」を用いて再現できます。

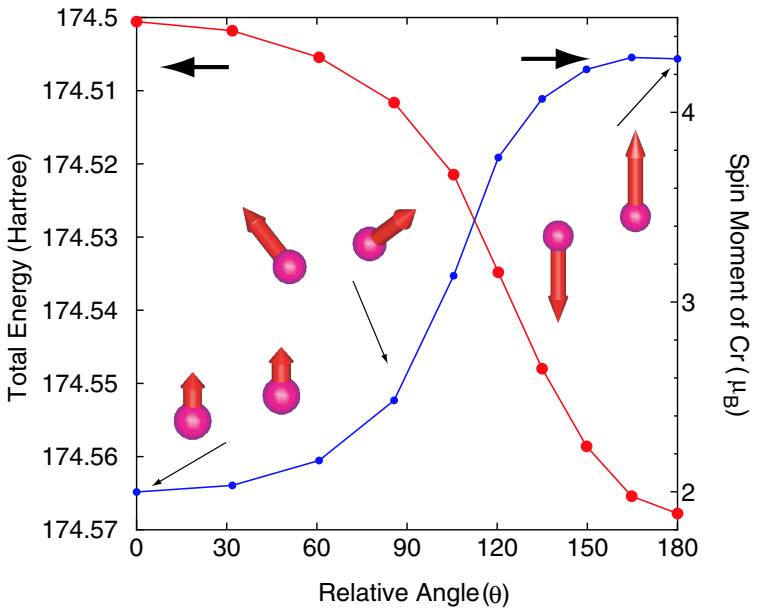


図 28: クロム 2 量体の全エネルギーと Cr 原子サイトの磁気モーメント。結合距離は 2.0 Å。入力ファイルは「work」ディレクトリ中の「Cr2_CNC.dat」。

34 Zeeman 項

スピンおよび軌道磁気モーメントに Zeeman 項を適用することができます。

34.1 スピン磁気モーメントに対する Zeeman 項

次のキーワードによって、スピン磁気モーメントに対する Zeeman 項が一様磁場との相互作用として導入できます。

```
scf.NC.Zeeman.Spin            on        # on|off, default=off
scf.NC.Mag.Field.Spin        100.0      # default=0.0(Tesla)
```

スピン磁気モーメントに対し Zeeman 項を導入する場合、キーワード「scf.NC.Zeeman.Spin」を「on」にします。一様磁場の強度は、キーワード「scf.NC.Mag.Field.Spin」で、テスラ単位で指定できます。この方法は制約条件付き DFT 法として拡張することが可能で、各原子毎に磁場の方向を指定することができます。スピン磁気モーメントに対する磁場の方位は、キーワード「Atoms.SpeciesAndCoordinates」によって次のように指定されます。

```
<Atoms.SpeciesAndCoordinates
 1  Sc  0.000  0.000  0.000   6.6 4.4  10.0 50.0  160.0 20.0  1  on
 2  Sc  2.000  0.000  0.000   6.6 4.4  80.0 50.0  160.0 20.0  1  on
Atoms.SpeciesAndCoordinates>
```

スピン磁気モーメントに対する磁場の方位を指定するために、第 8 列と 9 列で Euler 角 (θ、ϕ) を指定します。第 12 列は制約条件に対するスイッチです。「1」は制約条件有り、「0」は無しを意味しています。各原子サイトに対して印加磁場の方位を制御できるため、この方法はノンコリニアスピン構造を研究する手段を提供します。キーワード「scf.NC.Zeeman.Spin」と「scf.Constraint.NC.Spin」は互いに両立できないことに注意してください。したがって、「scf.NC.Zeeman.Spin」がオンの場合、キーワード「scf.Constraint.NC.Spin」は次のようにオフにしなければなりません。

```
scf.Constraint.NC.Spin        off       # on|off, default=off
```

この章で説明したスピン磁気モーメントの方位に対する Zeeman 項と、前章で説明した制約法は、どちらもスピン方位を制御する方法ですが、Zeeman 項を付加した場合には、スピン磁気モーメントの大きさが増加する可能性があることに注意して下さい。

34.2 軌道磁気モーメントに対する Zeeman 項

次のキーワードによって、軌道磁気モーメントに対する Zeeman 項が一様磁場との相互作用として導入できます。

```
scf.NC.Zeeman.Orbital         on        # on|off, default=off
scf.NC.Mag.Field.Orbital     100.0      # default=0.0(Tesla)
```

軌道磁気モーメントに対して Zeeman 項を組み込む場合、キーワード「scf.NC.Zeeman.Orbital」を「on」にします。一様磁場の強度は、キーワード「scf.NC.Mag.Field.Orbital」で、テスラ単位で指定できます。この方法は制約条件付き DFT 法として拡張することが可能で、各原子毎に磁場の方向を指定することができます。軌道磁気モーメントに対する磁場の方位は、キーワード「Atoms.SpeciesAndCoordinates」によって次のように指定されます。

```
<Atoms.SpeciesAndCoordinates
 1  Sc  0.000  0.000  0.000   6.6 4.4  10.0 50.0  160.0 20.0  1  on
 2  Sc  2.000  0.000  0.000   6.6 4.4  80.0 50.0  160.0 20.0  1  on
Atoms.SpeciesAndCoordinates>
```

軌道磁気モーメントに対する磁場の方位を指定するために、第 10 列と 11 列で Euler 角 (θ、ϕ) を指定します。第 12 列は制約条件に対するスイッチです。「1」は制約条件有り、「0」は無しを意味しています。各原子サイトに対して印加磁場の方位を制御できるため、この方法はノンコリニアスピン構造を研究する手段を提供します。また軌道磁気モーメントに対する磁場の方位はスピン磁気モーメントの場合と同じである必要はありません。この機能を利用し、様々な磁気構造を計算することが可能です。

35 Berry 位相による巨視的分極の計算

Berry 位相の方法を用いて、バルクの巨視的電気分極を計算することができます [12]。例として、塩化ナトリウム中のナトリウム原子の Born 有効電荷を巨視的電気分極から計算する手順を説明します。

(1) SCF 計算

最初に、「work」ディレクトリ中にある入力ファイル「NaCl.dat」を用いて、通常の SCF 計算を実行します。この際に、キーワード「HS.fileout」をオンにします。

```
HS.fileout              on     # on|off, default=off
```

SCF 計算が正常に完了すると、「work」ディレクトリ中に出力ファイル「nacl.scfout」が生成されます。

(2) 巨視的分極の計算

巨視的分極はポストプロセスのプログラム「polB」を用いて計算します。この際に「nacl.scfout」が入力データとなります。まず「source」ディレクトリにおいて、「polB」を次のようにコンパイルします。

```
% make polB
```

コンパイルが正常に完了すると、「work」ディレクトリ中に実行形式ファイル「polB」が生成されます。次に、「work」ディレクトリに移動し、次のように実行します。

```
% polB nacl.scfout
```
もしくは
```
% polB nacl.scfout < in > out
```

後者の場合、テキストファイル「in」には次のデータが保存されています。

```
9 9 9
1 1 1
```

前者の場合、次のように会話形式でプログラムから質問されます。

```
*****************************************************************
*****************************************************************
 polB:
 code for calculating the electric polarization of bulk systems
 Copyright (C), 2006-2007, Fumiyuki Ishii and Taisuke Ozaki
 This is free software, and you are welcome to
 redistribute it under the constitution of the GNU-GPL.
*****************************************************************
*****************************************************************
```

```
Read the scfout file (nacl.scfout)
 Previous eigenvalue solver = Band
 atomnum                    = 2
 ChemP                      = -0.156250000000 (Hartree)
 E_Temp                     = 300.000000000000 (K)
 Total_SpinS                =  0.000000000000 (K)
 Spin treatment             = collinear spin-unpolarized

 r-space primitive vector (Bohr)
  tv1=  0.000000   5.319579   5.319579
  tv2=  5.319579   0.000000   5.319579
  tv3=  5.319579   5.319579   0.000000
 k-space primitive vector (Bohr^-1)
  rtv1= -0.590572   0.590572   0.590572
  rtv2=  0.590572  -0.590572   0.590572
  rtv3=  0.590572   0.590572  -0.590572

  Cell_Volume=301.065992 (Bohr^3)

 Specify the number of grids to discretize reciprocal a-, b-, and c-vectors
 (e.g 2 4 3) 9 9 9
   k1    0.00000    0.11111    0.22222    0.33333    0.44444  ...
   k2    0.00000    0.11111    0.22222    0.33333    0.44444  ...
   k3    0.00000    0.11111    0.22222    0.33333    0.44444  ...

 Specify the direction of polarization as reciprocal a-, b-, and c-vectors
 (e.g 0 0 1 ) 1 1 1
```

逆格子ベクトルの離散化グリッド数と分極の方向を指定した後に、計算が以下のように進行します。

```
calculating the polarization along the a-axis ....
The number of strings for Berry phase : AB mesh=81

  calculating the polarization along the a-axis ....   1/  82
  calculating the polarization along the a-axis ....   2/  82
  .....
  ...

*****************************************************
              Electric dipole  (Debye) : Berry phase
```

```
*********************************************************

 Absolute dipole moment      163.93373639

              Background        Core              Electron          Total

 Dx          -0.00000000       94.64718996       -0.00000338       94.64718658
 Dy          -0.00000000       94.64718996       -0.00000283       94.64718713
 Dz          -0.00000000       94.64718996       -0.00000317       94.64718679


*************************************************************
             Electric polarization (muC/cm^2) : Berry phase
*************************************************************

              Background        Core              Electron          Total

 Px          -0.00000000      707.66166752       -0.00002529      707.66164223
 Py          -0.00000000      707.66166752       -0.00002118      707.66164633
 Pz          -0.00000000      707.66166752       -0.00002371      707.66164381


Elapsed time = 77.772559 (s) for myid=  0
```

V_c を単位セルの体積、e を素電荷、Δu_β を β 座標方向の変位、ΔP_α を α 座標方向の巨視的分極の変化とすると、Born 有効電荷 $Z^*_{\alpha\beta}$ は次のようにテンソルで定義されます。

$$Z^*_{\alpha\beta} = \frac{V_c}{|e|}\frac{\Delta P_\alpha}{\Delta u_\beta}$$

上記の表式に従い、Born 有効電荷を計算する際には、ナトリウム原子の x、y、z 座標を変化させながら、上の手順を少なくとも 2 回もしくは 3 回実行します。例えば x 座標方向を変位させて分極を計算すると、次の結果が得られます。

```
Px = 94.39497736 (Debye/unit cell) at x= -0.05 (Ang)
Px = 94.64718658 (Debye/unit cell) at x=  0.0  (Ang)
Px = 94.89939513 (Debye/unit cell) at x=  0.05 (Ang)
```

したがって、ナトリウム原子の Born 有効電荷は以下のように計算されます。

$$\begin{aligned} Z^*_{xx} &= \frac{(94.89939513 - 94.39497736)/(2.54174776)}{0.1/0.529177} \\ &= 1.050 \end{aligned}$$

表 5: Berry 位相による巨視的分極の計算から見積もられた NaCl 中の Na の Born 有効電荷。入力ファイルは「work」ディレクトリ中の「NaCl.dat」。比較のため、他の計算値 (FD: 参考文献 [72]) と実験値 (参考文献 [73]) も併記。

	OpenMX	FD	Expt.
Z^*	1.05	1.09	1.12

塩化ナトリウム固体では、Born 電荷のテンソルの非対角項はゼロ、そして $Z^*_{xx} = Z^*_{yy} = Z^*_{zz}$ となります。表 5 に、ここでの計算値と他の計算値 [72] および実験値 [73] との比較を示します。計算値と実験値が良く一致していることが分かります。巨視的分極の計算はコリニアおよびノンコリニア DFT 法の両者に対してサポートされています。また MPI プロセス数が系の原子数を上回る場合でも、プログラム「polB」は効率的な並列計算が可能です。

36　交換結合パラメータ

Green 関数法を用いて、各原子サイトに局在したスピン間の有効相互作用を交換結合係数として評価することができます [14, 15]。この機能はコリニア DFT 法のクラスター計算とバンド計算に対してのみサポートされています。また OpenMX Ver. 3.7 では、ポストプロセス計算で使用するプログラム「jx」はバンド計算の場合にのみ MPI 並列化されています。クラスター計算の場合には、MPI 並列化が実装されていませんので注意して下さい。

次の二つのステップで、各原子サイトに局在するスピン間の交換結合係数を計算されます。

(1) SCF 計算

まず最初に通常の SCF 計算を実行します。ここでは例として、「work」ディレクトリ中の入力ファイル「Fe2.dat」を使用して、コリニア DFT 計算を実行します。この計算は鉄原子の 2 量体に対するものです。ただし SCF 計算の際に、次のキーワード「HS.fileout」を設定する必要があります。

```
HS.fileout                    on       # on|off, default=off
```

計算が正常に完了すると、「work」ディレクトリ中にファイル「fe2.scfout」が生成されます。

(2) 交換結合係数の計算

交換結合パラメータを計算するためにプログラムコード「jx」をコンパイルします。ディレクトリ「source」に移動し、次のようにコンパイルします。

```
% make jx
```

コンパイルが正常に終了すると、「work」ディレクトリ中に実行ファイル「jx」が生成されます。交換結合係数の計算するためにはプログラム「jx」を用いて、以下のように実行します。ここで「*.scfout」が入力データとなります。

```
% ./jx fe2.scfout
```

プログラムとの会話形式で交換結合係数の計算を進めていきます。

```
*******************************************************************
*******************************************************************
 jx: code for calculating an effective exchange coupling constant J
 Copyright (C), 2003, Myung Joon Han, Jaejun Yu, and Taisuke Ozaki
 This is free software, and you are welcome to
 redistribute it under the constitution of the GNU-GPL.
*******************************************************************
*******************************************************************
```

```
Read the scfout file (fe2.scfout)
 Previous eigenvalue solver = Cluster
 atomnum                    = 2
 ChemP                      = -0.108015991530 (Hartree)
 E_Temp                     = 600.000000000000 (K)

Evaluation of J based on cluster calculation
 Diagonalize the overlap matrix
 Diagonalize the Hamiltonian for spin= 0
 Diagonalize the Hamiltonian for spin= 1

 Specify two atoms (e.g 1 2, quit: 0 0)  1 2
 J_ij between 1th atom and 2th atom is 848.136902053845 cm^{-1}
 Specify two atoms (e.g 1 2, quit: 0 0)  2 1
 J_ij between 2th atom and 1th atom is 848.136902053844 cm^{-1}
 Specify two atoms (e.g 1 2, quit: 0 0)  0 0
```

交換結合係数を計算する二つの原子を指定して下さい。また「0 0」と打ち込むことで、プログラム「jx」が終了します。

37　光学伝導度

この機能にはプログラムバグがいくつかあります。修正したコードを今後リリースする予定です。

線形応答理論に基づき、光学伝導度を計算することができます [51]。OpenMX Ver. 3.7 は、この計算をコリニアクラスター計算に対してのみサポートしています。

次の二つのステップで、分子の光学伝導度が計算されます。

(1) SCF 計算

まず最初に通常の SCF 計算を実行します。ここでは例として、「work」ディレクトリ中の入力ファイル「Methane_OC.dat」を使用して、コリニアクラスター計算を実行します。この計算はメタン分子に対するものです。ただし SCF 計算の際に、キーワード「Dos.fileout」と「OpticalConductivity.fileout」を次のように設定します。

```
Dos.fileout                    on    # on|off, default=off
OpticalConductivity.fileout    on    # on|off, default=off
```

計算が正常に完了すると、「work」ディレクトリ中にファイル「met.optical」と「met.Dos.val」が生成されます。

(2) 光学伝導度の計算

光学伝導度を計算するためにプログラムコード「OpticalConductivityMain」をコンパイルします。ディレクトリ「source」に移動し、次のようにコンパイルします。

```
% make OpticalConductivityMain
```

コンパイルが正常に完了すると、「work」ディレクトリ中に実行ファイル「OpticaConductivityMain」が生成されます。光学伝導度の計算するためにはプログラム「OpticaConductivityMain」を用いて、以下のように実行します。ここで「*.optical」と「*.Dos.val」が入力データとなります。

```
% ./OpticalConductivityMain met.optical met.Dos.val met.optout
```

プログラムとの会話形式で光学伝導度の計算を進めていきます。

```
 # freqmax=100.000000
 # gaussian=0.036749
 freqmax (Hartree)=? 3
 freq mech=? 1000
```

出力ファイル「met.optout」中で第 2、3 および 4 列は、それぞれ周波数 (Hartree)、アップおよびダウンスピンに対する光学伝導度（任意単位）となります。

38　電気伝導計算

38.1　概要

非平衡グリーン関数法（non-equilibrium Green function method、 NEGF 法）に基づき、分子、ナノワイヤ、超格子構造などの電子に由来する電気伝導特性を計算することが可能です。電気伝導計算の機能はコリニアとノンコリニア計算のどちらにもサポートされています。その特徴と機能を以下に列挙します。

- ゼロおよび有限バイアス電圧下における二つの電極に接続した系の SCF 計算
- ゲートバイアス電圧下での SCF 計算
- LDA+U 法と併用が可能
- スピン依存の透過率と電流
- 電流に垂直な方向に対して **k** 分解された透過率と電流
- 電流-電圧曲線の計算
- 高精度・高効率な周回積分
- バイアス電圧効果の補間
- ゼロバイアス電圧下における周期系の迅速な透過率計算

各機能の実装の詳細については、文献 [54] を参照して下さい。まずコニリア計算の場合について、各機能の使用法を説明します。その後に、ノンコリニア計算に関して、補足説明します。

考察する系

OpenMX Ver. 3.7 の実装では、図 29(a) に示す系を NEGF 法により取り扱います。この系は左右の無限電極部分とそれに接続する中心部からなり、また **bc** 面上での二次元周期性が仮定されています。二次元周期性を考慮すると、この系は図 29(b) に示す Bloch 波数ベクトル **k** に依存する一次元問題に変換することができます。中心領域 C_0 と左右の領域 L_0 および R_0 との境界面近くの電子構造の緩和を考慮するために、OpenMX Ver. 3.7 の実装では、領域 $C(\equiv L_0|C_0|R_0)$ のグリーン関数が自己無撞着に決定されます。現実装においては、ユニットセルの **a** 軸が電子輸送の方向であると仮定されていることに留意してください。計算モデルの幾何構造を作成する際には、この仕様を守らなければなりません。これに関連して、「ステップ 1：リード線部の計算」の節も参照して下さい。

計算の流れ

NEGF 計算は次の三段階で実行します。

ステップ 1 → ステップ 2 → ステップ 3

各ステップを以下に説明します。

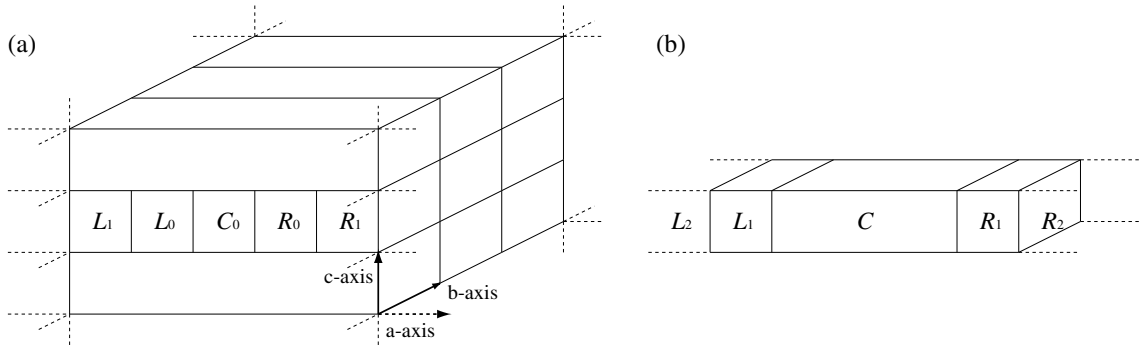


図 29: NEGF 法により取り扱う系の構成。**a** 軸方向の左右に無限電極が配置され、また **bc** 面上に二次元周期境界条件を仮定。(b) **bc** 面内での周期性を考慮することにより、(a) に示す構成から一次元化された系。C 領域は、C_0、L_0 および R_0 からなる拡張された中心領域を表す。

- **ステップ 1**

  左右の電極部分のバンド構造計算を、プログラムコード「openmx」を用いて実行。得られた計算結果は、ステップ 2 の NEGF 計算における電極部分に使用。

- **ステップ 2**

  図 29 に示した系に対して、ゼロまたは有限バイアス電圧下での NEGF 計算をプログラムコード「openmx」を用いて実行。この際に、ステップ 1 で計算した結果を電極部分の構築に使用する。

- **ステップ 3**

  ステップ 2 で得た結果を利用し、プログラムコード「TranMain」を用いて、透過率と電流を計算。

例：炭素鎖

最初の試みとして炭素鎖を例にとり、上述の三つのステップについて説明します。説明を始める前に、ステップ 3 で用いる「TranMain」を、「source」ディレクトリにおいて、次のようにコンパイルして下さい。

```
% make TranMain
```

コンパイルが成功すると「TranMain」の実行ファイルができますので、このファイルをユーザーの作業ディレクトリ、例えば「work」にコピーしてください。その後に次の三つの計算を実行して下さい。

ステップ 1

```
%./openmx Lead-Chain.dat | tee lead-chain.std
```

ステップ 1 の計算により、ファイル「negf-chain.hks」が生成します。

ステップ 2

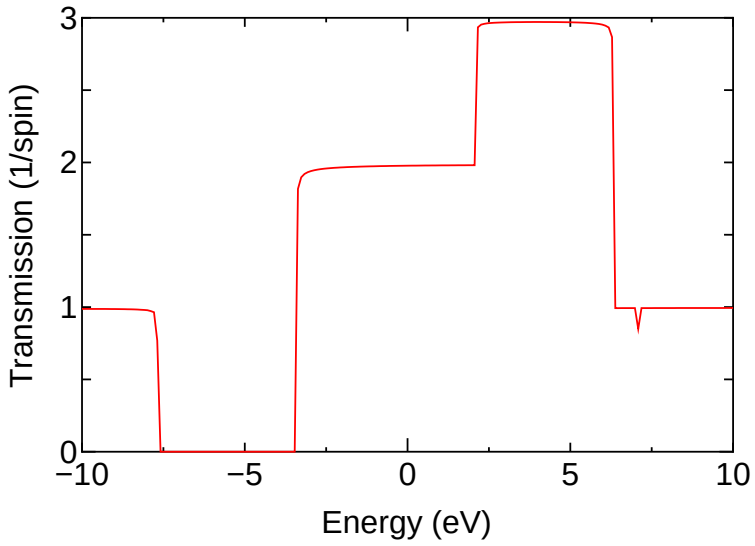


図 30: 炭素鎖の透過率のエネルギー依存性。エネルギーの原点は左電極の化学ポテンシャルに設定。

```
%./openmx NEGF-Chain.dat | tee negf-chain.std
```

ステップ 2 の計算により、ファイル「negf-chain.tranb」が生成します。

ステップ 3

```
%./TranMain NEGF-Chain.dat
```

ステップ 3 の計算により、「negf-chain.tran0_0」、「negf-chain.current」、「negf-chain.conductance」が生成します。

ディレクトリ「work/negf_example」中の入力ファイルを用いると、この計算を再現することができます。ステップ 3 の計算で得られた「negf-chain.tran0_0」の六列目を四列目の関数としてプロットすると、図 30 に示す透過率曲線が得られます。

38.2 ステップ 1：電極部分の計算

ステップ 1 の計算は、通常のバンド構造計算と同じものです。ただし入力ファイルに二つのキーワード「NEGF.output_hks」と「NEGF.filename.hks」を付け加えます。

```
NEGF.output_hks     on
NEGF.filename.hks   lead-chain.hks
```

上記のキーワードを付け加えることで、電極部分を構築するための情報がファイルに保存されます。ハミルトニアン行列要素、電子分布、および差電子 Hartree ポテンシャルが、キーワード「NEGF.filename.hks」で指定されるファイルに保存されます。上記の場合には、ファイル「lead-chain.hks」が生成されます。「*.hks」という拡張子のファイルは、ステップ 2 の計算で使用されることになります。現実装では電子

輸送の方向は **a** 軸方向に仮定されていますので、バンド構造計算においては、この仕様を念頭において電極モデルを構築する必要があります。しかし、ユーザー自身で構造を回転させる必要はありません。格子ベクトルの指定を変更するだけで、簡単に電子輸送の方向を適切に設定することが可能です。例えば、次の格子ベクトルにおいて、**a** 軸としてベクトル (0.0, 0.0, 10.0) を指定したい場合には

```
<Atoms.UnitVectors
  3.0  0.0  0.0
  0.0  3.0  0.0
  0.0  0.0 10.0
Atoms.UnitVectors>
```

以下のようにベクトルの順番を入れ替えるだけで、電子輸送の方向を適切に設定できます。

```
<Atoms.UnitVectors
  0.0  0.0 10.0
  3.0  0.0  0.0
  0.0  3.0  0.0
Atoms.UnitVectors>
```

この様に指定すれば、(0.0, 0.0, 10.0) の方向が電子輸送の方向になります。上の例で示したように、格子ベクトルの順序を変更する際には、キーワード「scf.Kgrid」も同様に変更しなければならないことに注意して下さい。

ステップ 2 の計算において、表面グリーン関数を用いることにより、電極部分の半無限性を考慮しています。そのため、ステップ 2 の計算では **a** 軸方向に関しては波数空間の離散化を導入することなしに、半無限性を取り扱うことができるようになります。**a** 軸方向の半無限性の取扱いに関し、ステップ 1 とステップ 2 での計算の整合性を保つために、ステップ 1 のバンド構造計算では **a** 軸方向に多数の **k** 点を割り当るようにして下さい。また、**bc** 面上の **k** 点数は、ステップ 1 とステップ 2 の計算において同一の値を使用して下さい。

38.3 ステップ 2：NEGF 計算

A. 左電極 | デバイス | 右電極の設定

図 29 に示した領域 L_0、C_0、R_0 は次のようにして設定できます。

中心領域 C_0 の幾何学的構造は、キーワード「Atoms.Number」および「Atoms.SpeciesAndCoordinates」により、以下のように指定します。

```
Atoms.Number         18
<Atoms.SpeciesAndCoordinates
  1  C  3.000  0.000  0.000  2.0 2.0
   .....
 18  C 28.500  0.000  0.000  2.0 2.0
```

```
Atoms.SpeciesAndCoordinates>
```

左側の電極部分 L_0 の幾何学的構造は、キーワード「LeftLeadAtoms.Number」および「LeftLeadAtoms.SpeciesAndCoordinates」により、以下のように指定します。

```
LeftLeadAtoms.Number          3
<LeftLeadAtoms.SpeciesAndCoordinates
  1  C -1.500  0.000  0.000  2.0 2.0
  2  C  0.000  0.000  0.000  2.0 2.0
  3  C  1.500  0.000  0.000  2.0 2.0
LeftLeadAtoms.SpeciesAndCoordinates>
```

右側の電極部分 R_0 の幾何学的構造は、キーワード「RightLeadAtoms.Number」および「RightLeadAtoms.SpeciesAndCoordinates」により、以下のように指定します。

```
RightLeadAtoms.Number         3
<RightLeadAtoms.SpeciesAndCoordinates
  1  C 30.000  0.000  0.000  2.0 2.0
  2  C 31.500  0.000  0.000  2.0 2.0
  3  C 33.000  0.000  0.000  2.0 2.0
RightLeadAtoms.SpeciesAndCoordinates>
```

ここで示した例は、上述の節で取り上げた炭素鎖の場合のものです。中心領域 C_0 は 18 個の炭素原子から構成され、左側の領域 L_0 と右側の領域 R_0 は、それぞれ 3 個の炭素原子から成り、すべての隣接原子間距離は 1.5 Å です。デバイス領域 C_0 と電極 L_0 および R_0 の幾何学的構造を設定することで、OpenMX は図 29 に示した拡張中心領域 $C(\equiv L_0|C_0|R_0)$ を構成します。中心領域 C_0 とその外側の $L_0(R_0)$ の境界付近での電子構造の緩和を考慮するために、拡張中心領域 C のグリーン関数が自己無撞着に決定されます。NEGF 法を用いて、拡張中心領域 C のグリーン関数を計算するために、次の二つの条件を満たすように計算モデルを構築する必要があります [54]。

1. 領域 C_0 における局在基底軌道 ϕ は、領域 L_0 および R_0 の局在基底軌道と重なるが、領域 L_1 および R_1 と重なることはない。

2. L_i (R_i) 領域における局在基底軌道 ϕ は、その最近接セル L_{i-1} (R_{i-1}) および L_{i+1} (R_{i+1}) より遠方のセル内の基底軌道と重なることはない。

OpenMX の基底関数は実空間において厳密に局在しているため [28, 29]、特定のカットオフ半径を持つ局在軌道を各領域に対して割り当てると、L_i および R_i のユニットセルの大きさを調整することにより、上記の二つの条件を常に満たすことができます。領域 L_0、C_0、R_0 の単位胞を指定する必要はありませんが、周期性を暗黙のうちに仮定していることに留意してください。半無限電極の構築は、ステップ 1 のバンド構造計算で用いた単位胞を利用して自動的に行われます。そのための情報はファイル「*.hks」に保存されています。また、図 29 に示した配置構造のため、左側と右側の電極の **bc** 面内の単位胞ベク

トルは一致していなければなりません。また拡張中心領域 C に対する **bc** 面内の単位胞ベクトルは、電極部分の単位胞ベクトルと同じであることが暗黙のうちに仮定されています。ユーザーは、こうした制約の枠内で、幾何構造を設定できます。

原子位置の指定で用いられる単位は、

```
Atoms.SpeciesAndCoordinates.Unit   Ang # Ang|AU
```

により指定されますが、NEGF 計算では「Ang」または「AU」のみに対応しています。「FRAC」での指定は出来ませんので、注意して下さい。

どの様に OpenMX が指定した幾何学的な配置構造を解析したのか確認するために、標準出力にその情報が記載されています。上述の炭素鎖の例では以下の情報が出力されています。

```
<TRAN_Calc_GridBound>


*******************************************************
The extended cell consists of Left0-Center-Right0.
The cells of left and right reads are connected as.
...|Left2|Left1|Left0-Center-Right0|Right1|Right2...

Each atom in the extended cell is assigned as follows:
where '12' and '2' mean that they are in 'Left0', and
'12' has overlap with atoms in the Left1,
and '13' and '3' mean that they are in 'Right0', and
'13' has overlap with atoms in the 'Right1', and also
'1' means atom in the 'Center'.
*******************************************************

Atom1  = 12 Atom2  = 12 Atom3  = 12 Atom4  =  1 Atom5  =  1 Atom6  =  1 Atom7  =  1
Atom8  =  1 Atom9  =  1 Atom10 =  1 Atom11 =  1 Atom12 =  1 Atom13 =  1 Atom14 =  1
Atom15 =  1 Atom16 =  1 Atom17 =  1 Atom18 =  1 Atom19 =  1 Atom20 =  1 Atom21 =  1
Atom22 = 13 Atom23 = 13 Atom24 = 13
```

$L_0|C_0|R_0$ からなる拡張された中心領域の原子には数値が割り当てられています。「12」および「2」は原子が L_0 に属し、「12」は L_1 内の原子と基底関数間の重なりを持つことを意味します。「13」および「3」は原子が R_0 に属し、「13」は R_1 にある原子と基底関数間の重なりを持つことを意味します。また「1」は原子が C_0 に属していることを意味します。この出力を調べることで、計算する系の幾何学的な構造が適切に構築されているか否かを確認することができます。

B. キーワード

ステップ 2 の NEGF 計算を実行するために、キーワード「scf.EigenvalueSolver」を次のように指定します。

```
scf.EigenvalueSolver       NEGF
```

NEGF 計算に関連するキーワードを以下に列挙します。

```
NEGF.filename.hks.l     lead-chain.hks
NEGF.filename.hks.r     lead-chain.hks

NEGF.Num.Poles              100       # defalut=150
NEGF.scf.Kgrid              1 1       # defalut=1 1

NEGF.bias.voltage           0.0       # default=0.0 (eV)
NEGF.bias.neq.im.energy     0.01      # default=0.01 (eV)
NEGF.bias.neq.energy.step   0.02      # default=0.02 (eV)
```

各キーワードの内容を以下に説明します。

```
NEGF.filename.hks.l     lead-chain.hks
NEGF.filename.hks.r     lead-chain.hks
```

電極部分の情報を含むファイルは、上記の二つのキーワードにより指定できます。「NEGF.filename.hks.l」は左側、「NEGF.filename.hks.r」は右側の電極部分に対応します。

```
NEGF.Num.Poles              100       # defalut=150
```

平衡密度行列は、周回積分法により評価します [54, 55]。この方法で用いる極の数は、キーワード「NEGF.Num.Poles」により指定します。

```
NEGF.scf.Kgrid              1 1       # defalut=1 1
```

逆格子ベクトル $\tilde{\mathbf{b}}$ および $\tilde{\mathbf{c}}$ を離散化するための **k** 点数を、キーワード「NEGF.scf.Kgrid」により指定します。**a** 軸方向には周期性を持っていないため、**a** 軸に対しては指定する必要がありません。

```
NEGF.scf.Iter.Band           6        # defalut=6
```

SCF 計算の最初の数ステップでは、**b** 軸および **c** 軸と同様に **a** 軸方向にも周期性を仮定することにより、通常の対角化法を用いた方が最終的な収束が加速されることが分かっています。自己無撞着な NEGF 法においてしばしば問題となる SCF 計算での収束困難性は、この方法によってかなり軽減します。通常の対角化法を使用する SCF ステップ数を、キーワード「NEGF.scf.Iter.Band」により指定します。「NEGF.scf.Iter.Band」で指定した SCF ステップ数までは通常の対角化法を用い、それ以降のステップにおいては、NEGF 法が適用されることになります。デフォルト値は 6 です。

```
NEGF.bias.voltage           0.0       # default=0.0 (eV)
```

キーワード「NEGF.bias.voltage」により、左右の電極間に印加するソース・ドレインバイアス電圧を eV の単位で指定します。この eV の単位は電圧に換算するとボルト (V) に対応しています。ソースとドレイン間の電位差のみが物理的に意味を持つため、電位の差であるソース・ドレインバイアス電圧をキーワードによって与えることになります。

```
NEGF.bias.neq.im.energy    0.01      # default=0.01 (eV)
NEGF.bias.neq.energy.step  0.02      # default=0.02 (eV)
```

有限のソース・ドレインバイアス電圧を印加すると、密度行列の一部は非平衡グリーン関数から計算されます。非平衡グリーン関数は複素平面上で一般に解析的ではありませんので、平衡グリーン関数で用いられた周回積分法は使用できません。現在の実装では、非平衡グリーン関数は微小虚部を付与した実軸上で単純な矩形求積法を用いて評価しています。この際、虚数部はキーワード「NEGF.bias.neq.im.energy」により指定します。またステップ幅はキーワード「NEGF.bias.neq.energy.step」により、eV の単位で与えます。通常はデフォルト値で十分な精度が確保できますが、収束性の詳細な議論は文献 [54] を参照して下さい。非平衡グリーン関数を評価する実軸上のエネルギー点数は標準出力およびファイル「*.out」で確かめることができます。「NEGF-Chain.dat」の例では、バイアス電圧が 0.5V の際には 120 個のエネルギー点が割り当てられており、以下のように標準出力から確認できます。

```
Intrinsic chemical potential (eV) of the leads
  Left lead:  -7.752843837400
  Right lead: -7.752843837400
  add voltage =  0.0000 (eV) to the  left lead: new ChemP (eV):  -7.7528
  add voltage =  0.5000 (eV) to the right lead: new ChemP (eV):  -7.2528


Parameters for the integration of the non-equilibrium part
  lower bound:           -8.706843837400 (eV)
  upper bound:           -6.298843837400 (eV)
  energy step:            0.020000000000 (eV)
  number of steps:           120
```

グリーン関数を評価するエネルギー点の総数は、非平衡グリーン関数を評価する実軸上のエネルギー点と、平衡グリーン関数を評価する極の数の総和で与えられます。計算時間は、エネルギー点の総数に比例することに留意して下さい。

```
NEGF.Poisson.Solver       FD     # FD|FFT, default=FD
```

NEGF 法では差電子密度に対する静電ポテンシャルを 2 次元 FFT+1 次元有限差分法（FD）[54] もしくは三次元 FFT 法（FFT）[56] のどちらかの方法で評価します。このポアソン・ソルバーの選択はキーワード「NEGF.Poisson.Solver」で行います。非極性系についてはどちらの方法も同様の静電ポテンシャルを与えますが、極性系については両者の差異は大きくなります。前者の FD では、電極部分と中心領域の境界条件が厳密に満されるため、理論的にはより正しい方法です。ここでの境界条件とは、電極部

分と中心領域の界面での静電ポテンシャルがステップ 1 の計算におけるポテンシャル値と同一であるというものです。SCF 収束性の観点からは、FD の収束性が悪く、一方、後者の FFT の方がより SCF 収束が容易であるという傾向が分かっています。デフォルトは FD です。

C. SCF の収束条件

NEGF 法では、キーワード「scf.criterion」により与えられる SCF の収束条件は残差ノルム「NormRD」に適用されます。NEGF 法以外の場合には、dUele が SCF の収束条件としてモニターされます。

D. ゲート・バイアス電圧

OpenMX Ver. 3.7 の実装では、次式で定義されるポテンシャルを加えることで、ゲート電圧 $V_{\rm g}(x)$ が取り扱われます。

$$V_{\rm g}(x) = V_{\rm g}^{(0)} \exp\left[-\left(\frac{x-x_{\rm c}}{d}\right)^8\right],$$

ここで、$V_{\rm g}^{(0)}$ はゲート電圧に対応する定数であり、キーワード「NEGF.gate.voltage」により指定できます。

```
NEGF.gate.voltage   1.0    # default=0.0 (in eV)
```

また x_c は領域 C_0 の中心位置、d は領域 C_0 の $\mathbf{a}$ 軸方向の単位ベクトルの長さです。式の形が示すように、印加されたゲート電圧は、主として拡張中心領域 C 内の領域 C_0 に作用します。このポテンシャルは、鏡像電荷によって生成されるポテンシャル形状に類似しています [57]。

E. 状態密度（DOS）

NEGF 計算において、状態密度 (DOS) は次のキーワードを設定することにより計算できます。

```
Dos.fileout                  on              # on|off, default=off
NEGF.Dos.energyrange     -15.0 25.0 5.0e-3  #default=-10.0 10.0 5.0e-3 (eV)
NEGF.Dos.energy.div        200               # default=200
NEGF.Dos.Kgrid             1 1               # default=1 1
```

NEGF 法において DOS を計算する際には、「Dos.fileout」を「on」にして下さい。また、DOS を計算するエネルギー範囲は、キーワード「NEGF.Dos.energyrange」により与えます。ここで、最初と二番目の数値はエネルギーの下限と上限で、三番目の数値は DOS を滑らかにするために用いる虚数値です。NEGF.Dos.energyrange」により指定されたエネルギー範囲は、キーワード「NEGF.Dos.energy.div」で指定した数値で分割され、そのエネルギー点上で DOS が計算されます。逆格子ベクトル $\tilde{\mathbf{b}}$ および $\tilde{\mathbf{c}}$ を離散化するための $\mathbf{k}$ 点のそれぞれの数を、キーワード「NEGF.Dos.Kgrid」により指定します。高精度に DOS を計算するために、「NEGF.Dos.Kgrid」で与える $\mathbf{k}$ 点数を、「NEGF.scf.Kgrid」で与える $\mathbf{k}$ 点数より大きくした方が良いでしょう。NEGF 計算が完了すると、「*.Dos.val」と「*.Dos.vec」の二つのファイルが生成します。その後の解析手順は、一般の場合と同様です。また、エネルギーの原点は**左側**の電極の化学ポテンシャルに設定されていることに留意してください。

38.4 ステップ 3: 透過率と電流

ステップ 2 とステップ 3 の計算を終了後、ステップ 2 の計算で用いた入力ファイルに次のキーワードを付け加えることにより、透過率と電流が計算できます。

```
NEGF.tran.energyrange -10 10 1.0e-3  # default=-10.0 10.0 1.0e-3 (eV)
NEGF.tran.energydiv        200       # default=200
NEGF.tran.Kgrid            1 1       # default= 1 1
```

キーワード「NEGF.tran.energyrange」により透過率を計算するエネルギー範囲を指定します。最初と二番目の数値は、エネルギーの下限値と上限値で、三番目の数値は透過率を滑らかにするための虚数値です。「NEGF.tran.energyrange」により指定されるエネルギー範囲は、キーワード「NEGF.tran.energydiv」により指定した数値で分割され、そのエネルギー点上で透過率が計算されます。逆格子ベクトル $\tilde{\mathbf{b}}$ および $\tilde{\mathbf{c}}$ を離散化するための $\mathbf{k}$ 点の数のそれぞれの値を、キーワード「NEGF.tran.Kgrid」により指定します。高精度に透過率を計算するために、「NEGF.tran.Kgrid」で与える $\mathbf{k}$ 点数を、「NEGF.scf.Kgrid」で与える $\mathbf{k}$ 点数より大きくした方が良いでしょう。

透過率と電流の計算は、プログラムコード「TranMain」により行われます。「source」ディレクトリにおいて次のようにコンパイルすることで、「TranMain」の実行ファイルが生成されます。

```
% make TranMain
```

コンパイルに成功すると「TranMain」の実行ファイルが生成されますので、このファイルをユーザーの作業ディレクトリ、例えば「work」にコピーしてください。「TranMain」を用いて、ステップ 3 の計算を次のように実行できます。

```
%./TranMain NEGF-Chain.dat

*********************************************************
*********************************************************
 Welcome to TranMain
 This is a post-processing code of OpenMX to calculate
 electronic transmission and current.
 Copyright (C), 2002-2013, H.Kino and T.Ozaki
 TranMain comes with ABSOLUTELY NO WARRANTY.
 This is free software, and you are welcome to
 redistribute it under the constitution of the GNU-GPL.
*********************************************************
*********************************************************


Chemical potentials used in the SCF calculation
  Left lead:  -7.752843837400 (eV)
```

```
  Right lead: -7.752843837400 (eV)
NEGF.current.energy.step 1.0000e-02 seems to be large for the calculation ....
The recommended Tran.current.energy.step is 0.0000e+00 (eV).

Parameters for the calculation of the current
  lower bound:    -7.752843837400 (eV)
  upper bound:    -7.752843837400 (eV)
  energy step:     0.010000000000 (eV)
  imginary energy  0.001000000000 (eV)
  number of steps:   0

  calculating...

  myid0= 0 i2= 0 i3= 0  k2=  0.0000 k3= -0.0000

Transmission:  files

  ./negf-chain.tran0_0

Current:  file

  ./negf-chain.current


Conductance:  file

  ./negf-chain.conductance
```

計算の終了後に、上記の例では三つのファイル「negf-chain.tran0_0」、「negf-chain.current」、「negf-chain.conductance」が生成されます。

- *.tran#_%

  このファイルはアップとダウンのスピン状態に対する透過率を保存します。4 番目の列は、**左側**リード線部の化学ポテンシャルに対する相対的なエネルギーで、6 番目と 8 番目の列は、それぞれアップとダウンスピン状態に対する透過率です。「NEGF.tran.Kgrid」により与えられる **k** 点の数を多く取ると、ファイル拡張子に「#」と「%」の異なる組を持つファイルが、各 **k** 点について生成されます。ファイルの数字と **k** 点との対応はファイル内で見られます。

- *.current

  このファイルには、アップスピン状態とダウンスピン状態に対する **k** 分解された電流およびその平均値がアンペアの単位で保存されています。

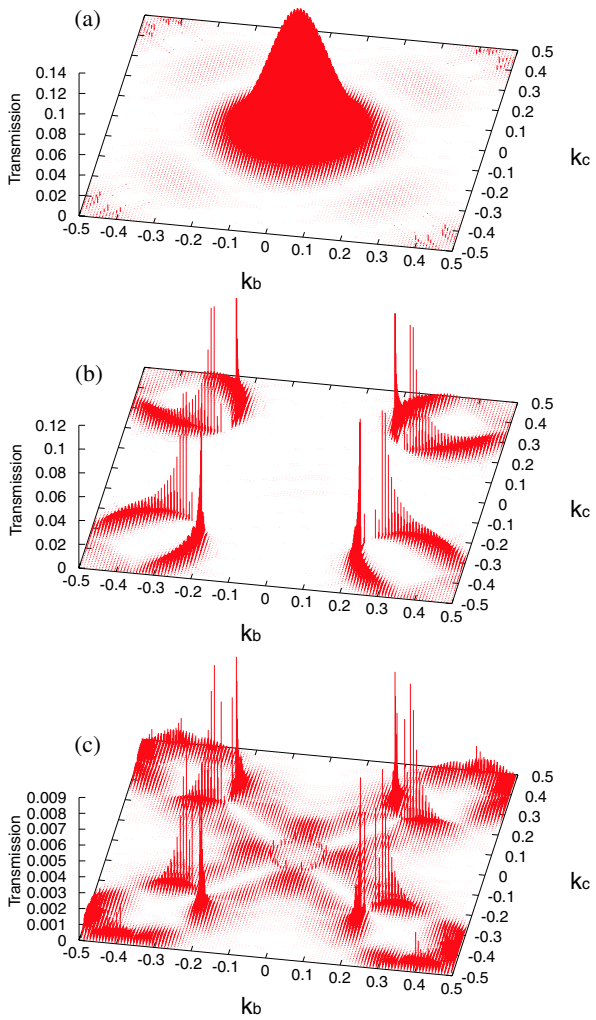


図 31: Fe|MgO|Fe の化学ポテンシャルでの **k** 分解透過率。(a) 平行スピン配置での多数スピン状態、(b) 平行スピン配置での少数スピン状態、(c) 反平行スピン配置でのスピン状態。各計算には、120×120 個の **k** 点を使用。

- *.conductance

  このファイルには、アップスピン状態とダウンスピン状態に対する 0K での **k** 分解されたコンダクタンスとその平均値が量子化コンダクタンスの単位（$G_0 \equiv \frac{e^2}{h}$）で保存されています。コンダクタンス G は、**左**電極の化学ポテンシャル μ_L での透過率 T に、次のように比例しています。

$$G = \frac{e^2}{h} T(\mu_L)$$

一例として、ファイル「*.conductance」を用いて作成された Fe|MgO|Fe 構造の **k** 分解透過率を図 31 に示します。

38.5 ゼロバイアス下における周期系

bc 面の周期性だけでなく **a** 軸方向にも周期性を持った系の透過率は容易に計算可能です。3 次元の周期性を持つ系の場合には、グリーン関数法を用いることなく、通常のバンド構造計算によって得られた

ハミルトニアンを利用し、ゼロバイアス電圧下での透過率が計算できます。この方法により、様々な幾何構造や磁気構造を持った系の輸送特性を低い計算コストで調べることが可能となります。とりわけ超格子構造などの計算には有用でしょう。この計算は、ステップ 1 のバンド構造計算において、キーワード「NEGF.Output.for.TranMain」を付け加えることで実行可能です。

```
NEGF.Output.for.TranMain    on
```

ステップ 1 の計算が正常に終了すると、ステップ 3 の計算に利用可能なファイル「*.tranb」を生成されます。つまり、ステップ 2 の計算が省略されることになります。

38.6 バイアス電圧効果の補間法

大規模な系の SCF 計算を各バイアス電圧において実行するのは非常に計算時間を要します。バイアス電圧効果の補間法を用いて NEGF 法による計算コストを低減することが可能です。次の手順により補間を行います。(i) 対象とするバイアス電圧領域から選択した 2、3 のバイアス電圧についての SCF 計算。(ii) 透過率および電流を計算するとき、中心散乱領域と右側電極部分に対するハミルトニアンのブロック要素、$H^{(\mathbf{k})}_{\sigma,C}$ と $H^{(\mathbf{k})}_{\sigma,R}$、さらに化学ポテンシャル μ_R について次のような線形補間を行う。

$$
\begin{aligned}
H^{(\mathbf{k})}_{\sigma,C} &= \lambda H^{(\mathbf{k},1)}_{\sigma,C} + (1-\lambda) H^{(\mathbf{k},2)}_{\sigma,C}, \\
H^{(\mathbf{k})}_{\sigma,R} &= \lambda H^{(\mathbf{k},1)}_{\sigma,R} + (1-\lambda) H^{(\mathbf{k},2)}_{\sigma,R}, \\
\mu_R &= \lambda \mu^{(1)}_R + (1-\lambda) \mu^{(2)}_R,
\end{aligned}
$$

ここで、上付きの添字 1 および 2 は、事前に SCF 計算を行ったバイアス電圧において、計算された、もしくは使用した量であることを意味します。補間の精度を保証するために、一般に内挿補間を行うべきであり、したがって λ は 0 から 1 の範囲の値に設定するべきです。

ステップ 3 の計算において、入力ファイルに次のキーワードを加えることにより補間を行います。

```
NEGF.tran.interpolate          on                # default=off, on|off
NEGF.tran.interpolate.file1  c1-negf-0.5.tranb
NEGF.tran.interpolate.file2  c1-negf-1.0.tranb
NEGF.tran.interpolate.coes     0.7 0.3           # default=1.0 0.0
```

補間を行う際には、キーワード「NEGF.tran.interpolate」を「on」に設定して下さい。上記の例では、キーワード「NEGF.tran.interpolate.file1」と「NEGF.tran.interpolate.file2」により指定されるファイル「cl-negf-0.5.tranb」と「cl-negf-1.0tranb」は、それぞれ 0.5 V および 1.0 V のバイアス電圧下での計算結果が保存されています。キーワード「NEGF.tran.interpolate.coes」により重み 0.7 と 0.3 が指定されていますので、$V = 0.7 \times 0.5 + 0.3 \times 1.0 = 0.65$ [V] における透過率と電流の値が本補間法により計算されます。

一次元炭素鎖の電流と透過率について、完全な SCF 計算と補間法の比較を図 32(a) および (b) に示します。補間法での計算では、0V、0.5V および 1.0V の三つのバイアス電圧において SCF 計算を行い、他のバイアス電圧における結果は補間で求めました。比較のために、補間法を用いずに完全な SCF 計算か

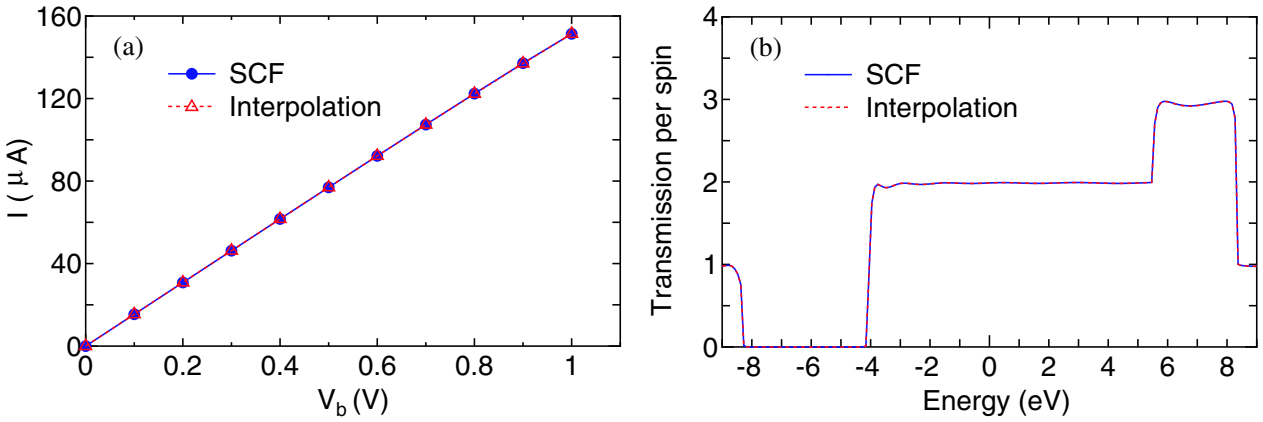


図 32: (a) SCF 計算（実線）と補間法（点線）により計算した一次元炭素鎖の電流、(b) SCF 計算（実線）と補間法（点線）により計算した 0.3V のバイアス電圧下での一次元炭素鎖の透過率。密度行列における非平衡成分の積分には虚数部 0.01 と格子間隔 0.01eV を使用。

ら求めた電流値も示しました。図 32 から、この簡易的な補間法は、電流および透過率のどちらもにも非常に正確な結果を与えていることが確認できます。補間の際の SCF 計算で用いるバイアス電圧の適切な選択は系に依存しますが、この結果は、バイアス電圧の効果を計算精度を保ちつつ補間するために、本方法が非常に有用であるということを示唆しています。

38.7 NEGF の並列化

NEGF 計算には MPI を用いた並列化が実装されています。MPI 並列化に加えて、ACML（AMD Core Math Library）または MKL（Math Kernel Library）を用いると、グリーン関数を評価するときの行列乗算および逆行列計算を OpenMP により並列化することもできます。この場合、OpenMP/MPI によるハイブリッド並列化を実行可能で、さらに計算時間を短縮することができます。並列計算の実行方法は以前に述べたものと全く同じです。

図 33 に、NEGF 計算における OpenMP/MPI ハイブリッド並列の速度向上比を示します。これは 0.5 eV の有限バイアス電圧下にある 8-ジグザグ型グラフェンナノリボン（ZGNR）の密度行列の計算を行った場合の結果です。密度行列計算には 197 個のエネルギー点（101 個は平衡密度行列、96 個は非平衡密度行列）を用いています。**k** 点サンプリングには Γ 点のみを用い、スピン分極計算を行いました。エネルギー点、**k** 点、スピンの三つのループに関して 394 個の組み合わせを MPI により並列化しました。1 スレッドに相当するフラット MPI 並列化の速度向上比は 64 プロセスまで順当に向上しています。さらに、2 および 4 スレッドに相当するハイブリッド並列化により速度向上比が大きく改善されていることが分かります。64 個のプロセスと 4 スレッドに相当する 64 個のクアッドコア・プロセッサを完全に用いた場合、速度向上比は約 140 であり、NEGF 法のスケーラビリティが良いことを実証しています。詳細については文献 [54] を参照して下さい。OpenMX Ver. 3.7 では、MPI 並列化におけるプロセス数が原子数を超えても効率的に並列化が実行されることにも留意して下さい。

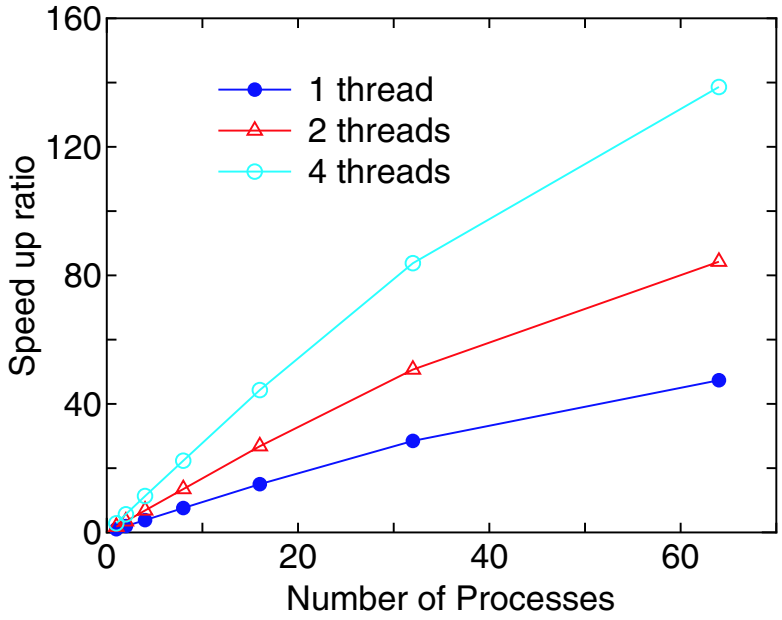


図 33: 8-ジグザググラフェンナノリボン（ZGNR）の密度行列の計算における OpenMP/MPI ハイブリッド並列の速度向上比。速度向上比は、T_1 および T_p をシングルコア計算および並列計算の経過時間とするとき、T_1/T_p により定義。**プロセス**および**スレッド**は MPI および OpenMP の並列計算で使用したコア数。並列計算は AMD opteron クワッドコア・プロセッサ（2.3 GHz）を搭載する CRAY-XT5 マシン上で実行。

38.8 ノンコリニア DFT 法に対する NEGF 法

OpenMX Ver. 3.7 は、ノンコリニア DFT 法と組み合わせた NEGF 法に対応しています。ノンコリニア DFT 法の全ての機能と NEGF 法は整合性を持った実装となっています。スピン-軌道相互作用、DFT+U 法、そしてスピン磁気モーメントおよび軌道磁気モーメントの方位に対する制約法の全ての機能が NEGF 法の実装と適合しています。したがって、らせん状磁気構造を持つ磁区を介した電子輸送など、広範な問題を取り扱うことが可能であると期待されます。NEGF 計算の実行の方法は、基本的にコリニア DFT 法の場合と同一です。コリニア計算とノンコリニア計算の唯一の違いは、ステップ 3 の計算がプログラム・コード「TranMain_NC」により実行されることです。ディレクトリ「source」中で、以下のようにコンパイルすることで実行ファイル「TranMain_NC」が生成されます。

```
% make TranMain_NC
```

コリニア計算と比較し、ステップ 3 の使用方法には大きな違いはありません。

ノンコリニア DFT 法と組み合わせた NEGF 計算の一例として、ジグザグ型グラフェン・ナノリボンの透過率の計算結果を図 34 に示します。ジグザグ端におけるスピン磁気モーメントは、左側および右側の電極部分のそれぞれにおいて、上方向および右方向に整列しています。電極部分の計算は、ステップ 1 においてスピン磁気モーメントの制約法を用いてノンコリニアバンド構造計算により行いました。次にステップ 2 の計算では制約条件を課しませんでした。ステップ 2 の SCF 計算の結果、図 34(a) に示すように中心部分のスピン方向は徐々に回転していくことが分かります。この計算は、「work/negf_example」ディレクトリに保存されている入力ファイル「Lead-L-8ZGNR-NC.dat」、「Lead-R-8ZGNER-NC.dat」、「NGEF-8ZGNR-NC.dat」により再現できます。また、同じディレクトリに金の一次元鎖のノンコリニア NEGF 計算の入力ファイルが保存されていますので、参考にして下さい。

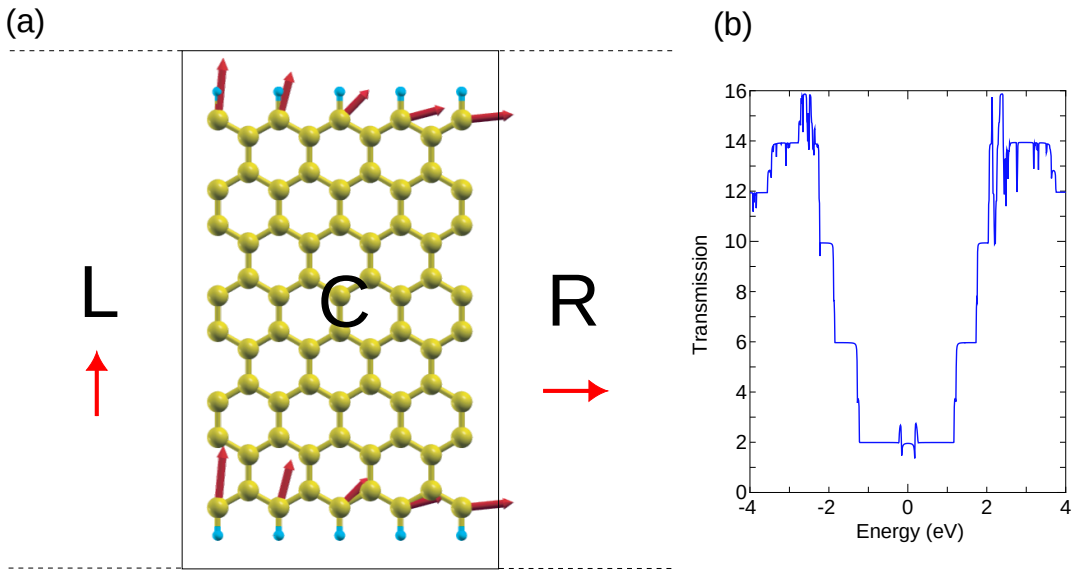


図 34: (a) 矢印で表したノンコリニアスピン方位を持つジグザグ型グラフェン・ナノリボン。矢印の長さは、スピン磁気モーメントの大きさに対応。ステップ 1 の計算において、スピン磁気モーメントの方位に対する制約法を適用。左側電極および右側電極において、ジグザグ端のスピン磁気モーメントがそれぞれ上方向および右方向となるように制約法を適用。(b) 図 34(a) に示すチャネル領域 C を通過する電子の透過率。

38.9 実例

NEGF 計算の参考例として、以下の５つの入力ファイルがディレクトリ「work/negf_example」中に保存されています。

- ゼロバイアス電圧下にある一次元炭素鎖

  ステップ 1: Lead-Chain.dat
  ステップ 2: NEGF-Chain.dat

- ゼロバイアス電圧下にあるグラフェン・シート

  ステップ 1: Lead-Graphene.dat
  ステップ 2: NEGF-Graphene.dat

- 0.3V の有限バイアス電圧下にある反強磁性接合を持つ 8-ジグザグ型グラフェン・ナノリボン

  ステップ 1: Lead-L-8ZGNR.dat, Lead-R-8ZGNR.dat
  ステップ 2: NEGF-8ZGNR-0.3.dat

- ゼロバイアス電圧下にあるノンコリニア磁気接合を持つ 8-ジグザグ型グラフェン・ナノリボン

  ステップ 1: Lead-L-8ZGNR-NC.dat, Lead-R-8ZGNR-NC.dat
  ステップ 2: NEGF-8ZGNR-NC.dat

- ノンコリニア NEGF 法で計算したゼロバイアス電圧下にある金の一次元鎖

ステップ 1: Lead-Au-Chain-NC.dat
ステップ 2: NEGF-Au-Chain-NC.dat

38.10 NEGF の自動実行テスト

NEGF 計算に関連する機能が適切にインストールされていることを確認するために、NEGF 計算の自動実行テストを次のようにして行うことができます。

MPI 並列実行の場合

```
% mpirun -np 16 openmx -runtestNEGF
```

OpenMP/MPI 並列実行の場合

```
% mpirun -np 8 openmx -runtestNEGF -nt 2
```

このテスト計算において、OpenMX はステップ 1 およびステップ 2 の計算を含む五つのテストケースを実行し、「work/negf_example」に保存されている参照結果と比較します。比較結果（全エネルギーと力の絶対差異）は、「work」ディレクトリ内にファイル「runtestNEGF.result」として保存されます。参照結果は、2.6 GHz Xeon マシンの 16 MPI プロセスを用いて計算されたものです。絶対差異が少数点以下 7 桁以内であれば、インストールが正常であると判断されます。

39 最局在ワニエ関数

39.1 概要

OpenMX Ver. 3.7 を用いて最局在ワニエ関数（MLWF）を生成することが可能です [78, 79]。MLWF を計算するためのキーワードと設定について以下に説明します。キーワードの様式は、OpenMX で本来用いている形式に厳密に従っています。具体的な例として、この章ではダイヤモンド構造のシリコンに関する計算結果を示します。計算はディレクトリ「work/wf_example」中にある入力ファイル「Si.dat」を用いて、プログラムコード「openmx」を用いて再現できます。その他のポストプロセスコードは使用しません。通常の SCF 計算の収束解が得られた後に、再スタートファイルを用い、適切な MLWF が得られるまで、いくつかのパラメータを変更しながら以下に説明する手順を繰り返すことになります。

この機能を用いて発表を行う際には、文献 [58] を引用して下さい。

MLWF を生成する

MLWF を生成するために、キーワード「Wannier.Func.Calc」を「on」と明示的に設定して下さい。デフォルト値は「off」です。

```
Wannier.Func.Calc        on          #default off
```

生成する MLWF の数の指定

生成する MLWF の数を、キーワード「Wannier.Func.Num」により指定します。デフォルト値はありません。

```
Wannier.Func.Num       4          #no default
```

ブロッホ状態を選択するエネルギーウィンドウ

MLWF は、1 組のブロッホ状態から生成されます。固有エネルギーに対するエネルギー窓を指定することで、1 組のブロッホ状態が選択されます。参考文献 [79] に従って、二つのエネルギー窓を導入します。1 つは、外エネルギー窓とよばれ、下限を示す「Wannier.Outer.Window.Bottom」と上限を示す「Wannier.Outer.Window.Top」の 2 つのキーワードにより指定されます。もう 1 つは、内エネルギー窓で、下限を示す「Wannier.Inner.Window.Bottom」と上限を示す「Wannier.Inner.Window.Top」の 2 つの同様なキーワードにより指定されます。これらの 4 つの値はすべて、フェルミ準位からの相対的な値を eV 単位で与えます。内エネルギー窓は外エネルギー窓の領域内に設定する必要があります。内エネルギー窓を定義する下限値と上限値が等しいときには、内エネルギー窓は定義されていないことになり、計算には使用できません。外エネルギー窓にデフォルト値はなく、また内エネルギー窓の下限値と上限値のデフォルト値はともに 0.0 です。例えば、次の様に設定します。

```
Wannier.Outer.Window.Bottom   -14.0   #lower boundary of outer window, no default value
Wannier.Outer.Window.Top        0.0   #upper boundary of outer window, no default value
Wannier.Inner.Window.Bottom     0.0   #lower boundary of inner window, default value 0.0
Wannier.Inner.Window.Top        0.0   #upper boundary of outer window, default value 0.0
```

目的のバンドを含む 2 つのエネルギー窓を適切に設定するためには、MLWF の計算の前にバンド構造と状態密度の一方または両方を計算し、その範囲を事前に知っておく必要があります。

MLWF の局在化計算では再スタートファイルから重なり行列要素を取得し、計算を行います。重なり行列はある特定の外エネルギー窓に対して計算され、ファイルに保存されています。したがって再計算の際に外エネルギー窓を再定義する場合には、事前に設定した外エネルギー窓の範囲内での変更が可能です。また計算を再スタートする際には、内エネルギー窓は外エネルギー窓の範囲内で自由に変更可能です。MLWF の両エネルギー窓の依存性を確かめる際には、上述の制約の範疇でエネルギー窓を調節して下さい。計算の再スタートさせる場合には、本節 7 項目の「重なり行列の計算を省いた最適化の再スタート」を参照してください。

MLWF の最初の推定

キーワード「Wannier.Initial.Guess」を「on」または「off」と設定することにより、MLWF の初期推定をするか否かを選択できます。デフォルト値は「on」です。多くの場合に MLWF の初期推定を行うことで、スプレッド関数を最小化する際の収束性が改善し、また局所的な解への収束を避けることが可能になります。初期推定を行う際には、生成する MLWF と同数の局在関数の組を定義しなければなりません。外エネルギー窓の内部のブロッホ波動関数は、局在関数の組に射影されます。したがって、このような局在関数はプロジェクタとも呼びます。プロジェクタを指定するには、以下の手順が必要です。

A. プロジェクタに用いる局在関数の定義

擬原子軌道（PAO）をプロジェクタに用いますので、PAO の設定は基底関数の場合と同様です。ダイヤモンド構造のシリコンに関する設定例は、以下の様になります。

```
Species.Number          2

<Definition.of.Atomic.Species
  Si       Si7.0-s2p2d1    Si_CA11
  proj1    Si5.5-s1p1d1f1  Si_CA11
Definition.of.Atomic.Species>
```

この例では、Si の PAO をプロジェクタとして使用し、原子種「proj1」を上記のように定義します。キーワードの対「<Definition.of.Atomic.Species」および「Definition.of.Atomic.Species>」内で、1 行目の Si 原子に加えて、プロジェクタに関する原子種「proj1」を定義します。この原子種「proj1」は「Si5.5-s1p1d1f1」と擬ポテンシャル「Si_CA」で定義されます。ただし、この行で定義された擬ポテンシャルは実際の計算では使用されません。単にデータ入力の一貫性を保つために与えています。どのような PAO

もプロジェクタとして使用可能ですが、各 1-成分に対しては単一の動径軌道のみが指定可能であり、すべての場合に「s1p1d1f1」の指定を推奨します。

B. プロジェクタの軌道、中心位置、配向の指定

対となるキーワード「<Wannier.Initial.Projectos」および「Wannier.Initial.Projectos>」を用いて、プロジェクタ名、局在軌道関数、局在軌道の中心、軌道の方位を指定する局所的な z 軸および x 軸を指定します。設定例を以下に示します。

```
<Wannier.Initial.Projectors
   proj1-sp3   0.250  0.250  0.250   -1.0 0.0 0.0    0.0  0.0 -1.0
   proj1-sp3   0.000  0.000  0.000    0.0 0.0 1.0    1.0  0.0  0.0
Wannier.Initial.Projectors>
```

各行には次の項目が記載されています。例えば、第 1 行目では、原子種名「proj1」はキーワード対「Definition.of.Atomic.Species」で定義されています。プロジェクタ名と選択された軌道をつなぐためにハイフン「-」を用います。「sp3」は、この原子種の sp3 混成軌道が生成するワニエ関数の初期推定として使用されることを意味します （利用可能な軌道およびその混成軌道を表 6 に示します)。この混成軌道からなるプロジェクタは、次に続く 3 つの数「0.25 0.25 0.25」で与えられる位置を中心として配置されます。これらの数値は、キーワード「Wannier.Initial.Projectors.Unit」により定義される単位で与えられています （説明は後述します)。その次の 3 つの数から成る 2 組の値は、局所座標系の z 軸および x 軸の方向をそれぞれ定義しています。ここで、各軸は xyz 座標系の 3 つの成分により定義されるベクトルにより指定されます。この例の第 1 行目では、局所 z 軸は元の x 軸に対して逆向きの「-1.0 0.0 0.0」で表されるベクトルで定義され、局所 x 軸はもとの z 軸に対して逆向きの「0.0 0.0 -1.0」で表されるベクトルで定義されています。第 2 行目では、局所軸は元の座標系と同じです。

プロジェクタとして用いる軌道として、PAO 自身かまたはそれらの混成軌道も使用可能です。「sp3」により定義されるプロジェクタの総数は 4 個であること注意してください。同様に、「sp」および「sp2」により定義されるプロジェクタの総数は、それぞれ 2 個および 3 個です。サポートされている PAO およびそれらの混成軌道のリストは表 6 にまとめられています。このリストに記載されていないプロジェクタは使用出来ませんので注意して下さい。

プロジェクタは、単位胞内のどこにでも配置することができます。その位置の指定には、単位胞ベクトルに相対的な規格化座標（FRAC）か、原子単位（AU）もしくはオングストローム（ANG）の単位で表したデカルト座標が使用できます。その選択はキーワード「Wannier.Initial.Projectors.Unit」で行います。

```
 Wannier.Initial.Projectors.Unit     FRAC     #AU, ANG or FRAC
```

k 点グリッドおよび隣接する k 点を結ぶ b ベクトル

キーワード「Wannier.Kgrid」により、Monkhorst-Pack の **k** 点グリッドを指定します。デフォルトの設定はありません。**k** 空間における微分を計算する際に有限差分を用いるために、中心の **k** 点からの距

表 6: プロジェクタ用の軌道。軌道の方位は z 軸および x 軸を再定義することで回転可能。

Orbital name	Number of included projector	Description
s	1	s orbital from PAOs
p	3	p_x, p_y, p_z from PAOs
px	1	p_x from PAOs
py	1	p_y from PAOs
pz	1	p_z from PAOs
d	5	$d_{z^2}, d_{x^2-y^2}, d_{xy}, d_{xz}, d_{yz}$ from PAOs
dz2	1	d_{z^2} from PAOs
dx2-y2	1	$d_{x^2-y^2}$ from PAOs
dxy	1	d_{xy} from PAOs
dxz	1	d_{xy} from PAOs
dyz	1	d_{xy} from PAOs
f	7	$f_{z^3}, f_{xz^2}, f_{yz^2}, f_{zx^2}, f_{xyz}, f_{x^3-3xy^2}, f_{3yx^2-y^3}$ from PAOs
fz3	1	f_{z^3} from PAOs
fxz2	1	f_{xz^2} from PAOs
fyz2	1	f_{yz^2} from PAOs
fzx2	1	f_{zx^2} from PAOs
fxyz	1	f_{xyz} from PAOs
fx3-3xy2	1	$f_{x^3-3xy^2}$ from PAOs
f3yx2-y3	1	$f_{3yx^2-y^3}$ from PAOs
sp	2	Hybridization between s and px orbitals, including $\frac{1}{\sqrt{2}}(s+p_x)$ and $\frac{1}{\sqrt{2}}(s-p_x)$
sp2	3	Hybridization among s, px, and py orbitals, including $\frac{1}{\sqrt{3}}s-\frac{1}{\sqrt{6}}p_x+\frac{1}{\sqrt{2}}p_y$, $\frac{1}{\sqrt{3}}s-\frac{1}{\sqrt{6}}p_x-\frac{1}{\sqrt{2}}p_y$ and $\frac{1}{\sqrt{3}}s+\frac{2}{\sqrt{6}}p_x$
sp3	4	Hybridization among s, px, py and pz orbitals: $\frac{1}{\sqrt{2}}(s+p_x+p_y+p_z)$, $\frac{1}{\sqrt{2}}(s+p_x-p_y-p_z)$ $\frac{1}{\sqrt{2}}(s-p_x+p_y-p_z)$, $\frac{1}{\sqrt{2}}(s-p_x-p_y+p_z)$
sp3dz2	5	Hybridization among s, p_x, p_y, p_z and d_{z^2} orbitals: $\frac{1}{\sqrt{3}}s-\frac{1}{\sqrt{6}}p_x+\frac{1}{\sqrt{2}}p_y$, $\frac{1}{\sqrt{3}}s-\frac{1}{\sqrt{6}}p_x+\frac{1}{\sqrt{2}}p_y$, $\frac{1}{\sqrt{3}}s-\frac{2}{\sqrt{6}}p_x$ $\frac{1}{\sqrt{2}}p_z+\frac{1}{\sqrt{2}}d_{z^2}, -\frac{1}{\sqrt{2}}p_z+\frac{1}{\sqrt{2}}d_{z^2}$
sp3deg	6	Hybridization among s, p_x, p_y, p_z and $d_{z^2}, d_{x^2-y^2}$ orbitals: $\frac{1}{\sqrt{6}}s-\frac{1}{\sqrt{2}}p_x-\frac{1}{\sqrt{12}}d_{z^2}+\frac{1}{2}d_{x^2-y^2}$, $\frac{1}{\sqrt{6}}s+\frac{1}{\sqrt{2}}p_x-\frac{1}{\sqrt{12}}d_{z^2}+\frac{1}{2}d_{x^2-y^2}$, $\frac{1}{\sqrt{6}}s-\frac{1}{\sqrt{2}}p_y-\frac{1}{\sqrt{12}}d_{z^2}-\frac{1}{2}d_{x^2-y^2}$, $\frac{1}{\sqrt{6}}s+\frac{1}{\sqrt{2}}p_y-\frac{1}{\sqrt{12}}d_{z^2}-\frac{1}{2}d_{x^2-y^2}$, $\frac{1}{\sqrt{6}}s-\frac{1}{\sqrt{2}}p_z+\frac{1}{\sqrt{3}}d_{z^2}, \frac{1}{\sqrt{6}}s+\frac{1}{\sqrt{2}}p_z+\frac{1}{\sqrt{3}}d_{z^2}$

離に応じて殻 (shell) ごとに隣接 k 点を結ぶ **b** ベクトルを探索します。探索する殻の最大数は、キーワード「Wnnier.Maxshells」により指定します。デフォルト値は 12 で、適切な **b** ベクトルの組を見出すことが出来ない場合には、最大数を増やして下さい。単位格子ベクトル間で大きなアスペクト比を持つ系の場合、問題が起きることがあります。その場合にはエラーメッセージが表示されます。しかし通常は、最大数 12 で有効に機能します。「Wannier.Kgrid」の適切な設定もまた、**b** ベクトルを見つけるときに役立ちます。ここで、各逆格子ベクトルに対する離散化のグリッド間隔は、互いにほとんど同等でなければなりません。

```
Wannier.MaxShells          12          # default value is 12.
Wannier.Kgrid           8  8  8        # no default value
```

ワニエ関数の広がり (スプレッド) の最小化

もつれたバンド （entangled band）の場合 [79]、MLWF を見出すためには 2 つのステップが必要となります。最初のステップは、非孤立バンドのもつれを解くことによりスプレッド関数のゲージ不変部分を最小化することです。第 2 ステップは孤立バンドの場合と同様です [78]。ゲージ依存部分は、スプレッド関数の勾配に応じて、選択したブロッホ波動関数のユニタリー変換により最適化します。最初のステップでは、3 つのパラメータを用いて MLWF を生成するための自己無撞着ループを制御します。それらは、SCF ループの最大数「Wannier.Dis.SCF.Max.Steps」、収束基準「Wannier.Dis.Conv.Criterion」、そして入出力部分空間プロジェクタの混合を制御するパラメータ「Wannier.Dis.Mixing.Para」です。

```
Wannier.Dis.SCF.Max.Steps        2000        # default 200
Wannier.Dis.Conv.Criterion       1e-12       # default 1e-8
Wannier.Dis.Mixing.Para          0.5         # default value is 0.5
```

第 2 ステップにおいては、3 種類の最適化法が利用できます。1 つは最急降下法 （SD: steepest descent）で、2 つ目は共役傾斜法 （CG: conjugate gradient）です。3 つ目は、始めに SD 法を用いその後 CG 法に切り替えるハイブリッド法です。キーワード「Wannier.Minimizing.Scheme」で、どの方法を使うかを指定します。0 は簡易な SD 法、1 は CG 法、2 はハイブリッド法です。SD 法のステップ長をキーワード「Wannier.Minimizing.StepLength」により指定します。CG 法では、割線法 （secant method）を用いて最適ステップ長を求めます。最大割線ステップと初期ステップ長を、それぞれ「Wannier.Minimizing.Secant.Steps」と「Wannier.Minimizing.Secant.StepLength」により指定します。また最小化ステップの最大数および収束条件を、それぞれ「Wannier.Minimizing.Max.Steps」および「Wannier.Minimizing.Conv.Criterion」により指定します。

```
Wannier.Minimizing.Scheme             2     # default 0, 0=SD 1=CG 2=hybrid
Wannier.Minimizing.StepLength        2.0    # default 2.0
Wannier.Minimizing.Secant.Steps       5     # default 5
Wannier.Minimizing.Secant.StepLength 2.0    # default 2.0
Wannier.Minimizing.Conv.Criterion   1e-12   # default 1e-8
Wannier.Minimizing.Max.Steps         200    # default 200
```

ハイブリッド法でのSD法およびCG法の最大の最適化ステップ数は「Wannier.Minimizing.Max.Steps」で指定した値となります。

重なり行列の計算を省いた最適化の再スタート

一旦、重なり行列 $M_{mn}^{(\mathbf{k},\mathbf{b})}$ を計算し、ファイルに保存しておけば、再計算が容易に実行できます。事前に計算された重なり行列 $M_{mn}^{(\mathbf{k},\mathbf{b})}$ を利用し、再計算する場合には、キーワード「Wannier.Readin.Overlap.Matrix」を「on」と設定して下さい。

```
Wannier.Readin.Overlap.Matrix       on      # on|off, default is on
```

重なり行列の計算は時間を要しますので、このキーワードを指定することで、無駄な計算を省くことが可能となります。このキーワードが指定している場合にはプログラムコード「openMX」は、ファイルから固有エネルギーおよび固有状態と共に、重なり行列を読み取ります。外エネルギー窓と **k** 点グリッドは、保存された重なり行列と固有値の計算時に用いたものと同じでなければならないことに注意して下さい。読み込み時にそれらのパラメーターの一貫性がチェックされます。内エネルギー窓、MLWF の初期推定、また収束条件に関しては、最適化を再スタートする際に変更することができます。「Wannier.Readin.Overlap.Matrix」が「off」に設定されている場合には、重なり行列が計算されて自動的にファイルに保存されます。「System.Name」で定義された名前と「.mmn」という拡張子を持つファイルが生成されます。また固有エネルギーと固有状態も、拡張子「.eigen」を持ったファイルとして保存されます。

39.2 解析

補間バンド構造の表示

補間バンド構造を表示するためには、キーワード「Wannier.Interpolated.Bands」を「on」に設定してください。

```
Wannier.Interpolated.Bands              on    # on|off, default=off
```

k 経路と経路に沿ったサンプリング密度などの他の必要な設定は、OpenMX におけるバンド分散を表示するための設定と同様です。したがって、補間バンド構造を作成するために、キーワード「Band.dispersion」を「on」に設定しなければなりません。収束後、補間バンド分散データは「.Wannier_band」の拡張子を持つファイルに保存されます。このファイルは「.Band」ファイルと同じ形式です。一例として、ダイヤモンド構造における Si の補間バンド構造と元のバンド構造を図 35(a) に示します。

MLWF の描画

MLWF を描画するためには、キーワード「Wannier.Function.Plot」を「on」に設定して下さい。デフォルト値は「off」です。

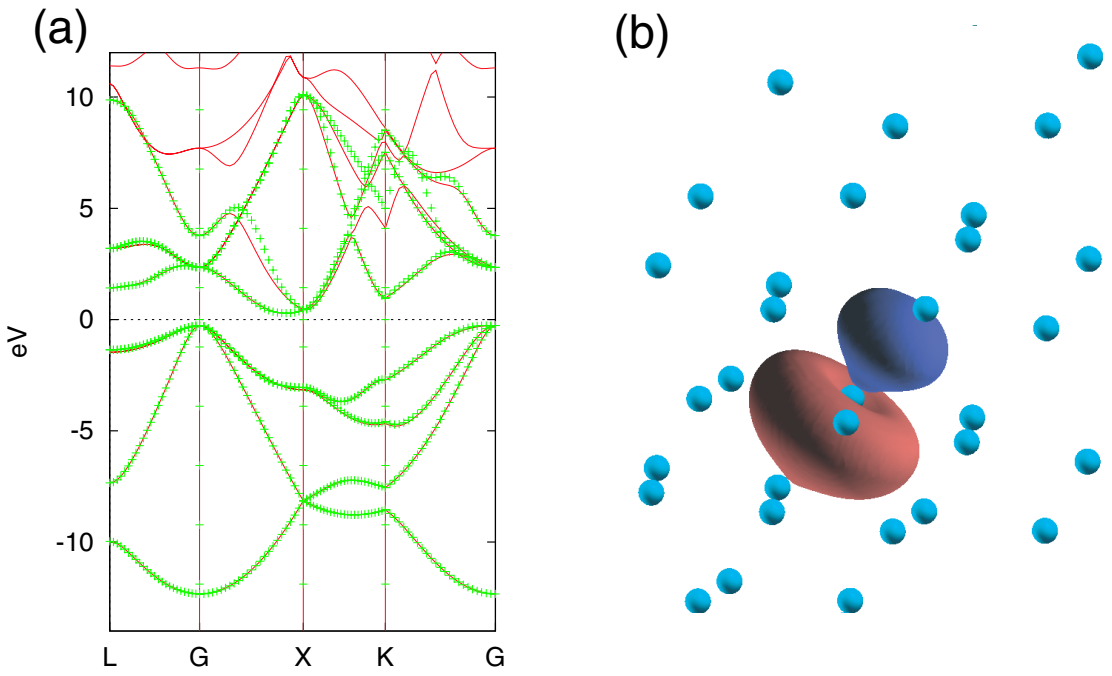


図 35: (a) ダイヤモンド構造における Si の補間バンド構造（シンボル線）と元のバンド構造（実線）との比較。(b) ダイヤモンド構造における Si のフェルミ準位近くの 4 つの価電子状態と 4 つの伝導状態からなる 8 つの MLWF の一つ。sp3 混成の初期推定から計算したもの。

```
Wannier.Function.Plot                    on          # default off
Wannier.Function.Plot.SuperCells       1 1 1         # default=0 0 0
```

キーワード「Wannier.Function.Plot」を「on」に設定すると、すべての MLWF を描画するためのファイルが生成されます。MLWF は、「.mlwfl_4_r.cube」の拡張子を持つ Gaussian Cube 形式のファイルで保存されます。このファイルは、HOMO や LUMO といった分子軌道ファイルと同じ形式で名前が付けられています。「.mlwfl」の後の数字はスピン・インデックスを、次の文字は MLWF のインデックスで、最後の文字「r」は real を、「i」は imaginary を表し、それぞれ MLWF の実部と虚数部という意味です。ユーザーは、MLWF を描画するスーパーセルのサイズを設定できます。キーワード「Wannier.Function.Plot.Supercells」によりそのサイズを指定します。上の例における「1 1 1」は、単位格子セルを中心に置き、a 軸、b 軸、c 軸方向のプラスおよびマイナス方向に 1 だけ拡大するという意味です。従って、この場合の MLWF は、27 個 ($= (1\times 2+1)\times(1\times 2+1)\times(1\times 2+1)$) のセルからなる拡張セルに描画されます。図 35(b) は、ダイヤモンド構造における Si のフェルミ準位近くの 4 つの価電子状態と 4 つの伝導状態からなる 8 つの収束した MLWF 内の一つを示したものです。

39.3 スプレッド関数の最適化過程の確認

最適化の過程は標準出力に書き出されます。以下の方法により、最適化の進行過程を確認して下さい。ここでは便宜上、標準出力はファイル「stdout.std」に保存されていると仮定します。以下で示す例は、「openmx*.*/work/wf_example」中の入力ファイル「Si.dat」を用いて、ユーザー各自が同じ計算を実行

することができます。

DISE

次の様にして、バンドのもつれをほどく処理（最適化の第 1 段階）の自己無撞着ループが確認できます。

```
% grep "DISE" stdout.std

|  Iter  | Omega_I (Angs^2) | Delta_I (Angs^2) |  ---> DISE
|     1  |   18.371525257652|   18.371525257652|  ---> DISE
|     2  |   17.955767336391|   -0.415757921261|  ---> DISE
|     3  |   17.659503060694|   -0.296264275698|  ---> DISE
|     4  |   17.454033576174|   -0.205469484520|  ---> DISE
|     5  |   17.311180447271|   -0.142853128902|  ---> DISE
|     6  |   17.210945408916|   -0.100235038355|  ---> DISE
|     7  |   17.139778800398|   -0.071166608519|  ---> DISE
|     8  |   17.088603102826|   -0.051175697572|  ---> DISE
|     9  |   17.051329329614|   -0.037273773211|  ---> DISE
|    10  |   17.023842837298|   -0.027486492316|  ---> DISE
........
.....
...
.
```

ここで、「Iter」は繰り返し数、「Omega_I」はスプレッド関数のゲージ不変部分、「Delta_I」は隣接する 2 つのステップ間の「Omega_I」の差をそれぞれ意味します。キーワード「Wannie.Dis.Conv.Criterion」に与えた収束条件が「Delta_I」に適用されます。

CONV

次の様にして、スプレッド関数のゲージ依存部分の最適化（最適化の第 2 段階）過程が確認できます。

```
% grep "CONV" stdout.std

Opt Step |Mode of Gradient|d_Omega_in_steps|     d_Omega    | (in Angs^2) ---> CONV
| SD    1 | 6.52434844E-01 | 5.41612774E-04 |-5.41340331E-04|  ---> CONV
| SD    2 | 6.51123660E-01 | 5.40524307E-04 |-5.40253165E-04|  ---> CONV
 .....
 .....
| SD  200 | 4.77499752E-01 | 3.96392019E-04 |-3.96271308E-04|  ---> CONV
```

```
|Opt Step |Mode of Gradient|     d_Omega     | (Angs^2) ---> CONV
| CG    1 | 8.61043764E-01 | -3.24716990E-01|  ---> CONV
 .....
 .....
| CG   58 | 1.67083857E-12 | -5.37225101E-13|  ---> CONV
| CG   59 | 5.44431651E-13 | -1.98972260E-13|  ---> CONV
************************************************************ ---> CONV
                  CONVERGENCE ACHIEVED !                     ---> CONV
************************************************************ ---> CONV
                  CONVERGENCE ACHIEVED !                     ---> SPRD
```

ここで、「Opt Step」はSD法またはCG法のどちらかにおける最適化ステップであり、「Mode of Gradient」は、スプレッド関数の勾配の絶対値です。SD法では、ゲージ依存スプレッド関数における隣接する2つのステップ間の差が2つの異なる方法で計算されます。その結果が「d_Omega_in_steps」および「d_Omega」です。「d_Omega_in_steps」は、

$$d\Omega = \epsilon \sum_{\mathbf{k}} ||G^{(\mathbf{k})}||^2,$$

であり、ϵ はステップ長、$G^{(\mathbf{k})}$ はスプレッド関数の勾配です。この式の詳細は参考文献 [78] に説明されています。一方、「d_Omega」は

$$d\Omega = \Omega^{(n+1)} - \Omega^{(n)},$$

により与えられます。ここで、n は反復（繰り返し）数です。CG法では、「d_Omega」のみを評価します。キーワード「Wannier.Minimizing.Conv.Criterion」で与えられた収束条件が、「Mode of Gradient」に適用されます。

SPRD

次の様にして、ワニエ関数の広がりの変化を確認できます。

```
% grep "SPRD" stdout.std

|Opt Step |     Omega_I    |     Omega_D    |     Omega_OD   |    Tot_Omega   | (in Angs^2) ---> SPRD
| SD    1 |   16.93053479  |    0.13727387  |    6.57748455  |   23.64529321  |  ---> SPRD
| SD    2 |   16.93053479  |    0.13724827  |    6.57696989  |   23.64475295  |  ---> SPRD
| SD    3 |   16.93053479  |    0.13722279  |    6.57645620  |   23.64421378  |  ---> SPRD
| SD    4 |   16.93053479  |    0.13719743  |    6.57594347  |   23.64367569  |  ---> SPRD
 .....
 .....
| SD  199 |   16.93053479  |    0.13399285  |    6.48989479  |   23.55442243  |  ---> SPRD
| SD  200 |   16.93053479  |    0.13398326  |    6.48950811  |   23.55402616  |  ---> SPRD
|Opt Step |     Omega_I    |     Omega_D    |     Omega_OD   |    Tot_Omega   | (Angs^2) ---> SPRD
| CG    1 |    16.93053479 |     0.15480701 |    6.14396737  |   23.22930917  |  ---> SPRD
| CG    2 |    16.93053479 |     0.17172507 |    5.87830203  |   22.98056189  |  ---> SPRD
```

```
| CG    3 |    16.93053479 |     0.17012089 |    5.78940789 |   22.89006357 |  ---> SPRD
 .....
 .....
| CG   57 |    16.93053479 |     0.16557875 |    5.73752928 |   22.83364282 |  ---> SPRD
| CG   58 |    16.93053479 |     0.16557876 |    5.73752928 |   22.83364282 |  ---> SPRD
| CG   59 |    16.93053479 |     0.16557876 |    5.73752928 |   22.83364282 |  ---> SPRD
********************************************************** ---> SPRD
                 CONVERGENCE ACHIEVED !                    ---> SPRD
********************************************************** ---> SPRD
```

ここで、「Opt Step」は SD 法または CG 法のどちらかにおける最適化ステップです。「Omega_I」はスプレッド関数のゲージ不変部分です。「Omega_D」はおよび「Omega_OD」は、それぞれゲージ依存の対角成分および非対角成分からの寄与です。「Tot_Omega」は、スプレッド関数の上記 3 つの成分の総和です。

CENT

次の様にしてワニエ関数中心の変化を確認できます。

```
% grep "CENT" stdout.std
WF   1 ( 1.14164289, 1.14164298, 1.14164266) |   2.95573380  --->CENT
WF   2 ( 1.55716251, 1.55716342, 1.14164203) |   2.95572597  --->CENT
WF   3 ( 1.55716191, 1.14164295, 1.55716190) |   2.95572978  --->CENT
WF   4 ( 1.14164389, 1.55716087, 1.55716055) |   2.95572957  --->CENT
WF   5 ( 0.20775982, 0.20775967, 0.20775893) |   2.95572677  --->CENT
WF   6 ( 0.20776045,-0.20775959,-0.20775914) |   2.95572605  --->CENT
WF   7 (-0.20775851, 0.20775981,-0.20775888) |   2.95572925  --->CENT
WF   8 (-0.20775787,-0.20775767, 0.20775933) |   2.95573335  --->CENT
Total Center ( 5.39761509, 5.39761243, 5.39760738) sum_spread  23.64583455 --->CENT
 SD     1 ------------------------------------------------------------------------> CENT
WF   1 ( 1.14164582, 1.14164592, 1.14164559) |   2.95566613  --->CENT
WF   2 ( 1.55715957, 1.55716049, 1.14164497) |   2.95565831  --->CENT
WF   3 ( 1.55715897, 1.14164588, 1.55715897) |   2.95566211  --->CENT
WF   4 ( 1.14164683, 1.55715794, 1.55715761) |   2.95566190  --->CENT
WF   5 ( 0.20775689, 0.20775673, 0.20775599) |   2.95565910  --->CENT
WF   6 ( 0.20775752,-0.20775666,-0.20775620) |   2.95565838  --->CENT
WF   7 (-0.20775558, 0.20775687,-0.20775594) |   2.95566158  --->CENT
WF   8 (-0.20775493,-0.20775474, 0.20775639) |   2.95566569  --->CENT
Total Center ( 5.39761509, 5.39761243, 5.39760738) sum_spread  23.64529321 --->CENT
 SD     2 ------------------------------------------------------------------------> CENT
 .....
 .....
 CG    59 ------------------------------------------------------------------------> CENT
WF   1 ( 1.14585349, 1.14584696, 1.14584386) |   2.85421846  --->CENT
WF   2 ( 1.55295615, 1.55294970, 1.14584792) |   2.85422167  --->CENT
WF   3 ( 1.55296133, 1.14584610, 1.55295139) |   2.85421070  --->CENT
WF   4 ( 1.14584053, 1.55296761, 1.55296391) |   2.85417080  --->CENT
WF   5 ( 0.20356211, 0.20355857, 0.20355600) |   2.85418933  --->CENT
WF   6 ( 0.20355119,-0.20355008,-0.20355192) |   2.85422458  --->CENT
```

```
WF   7 (-0.20355306, 0.20355395,-0.20355905) |   2.85420611  --->CENT
WF   8 (-0.20355603,-0.20356000, 0.20355520) |   2.85420117  --->CENT
Total Center ( 5.39761571, 5.39761281, 5.39760730) sum_spread  22.83364282 --->CENT
```

「SD」または「CG」で始まる行に、最適化法とそのステップ数が示されています。「WF」で始まる行は、各ワニエ関数の中心の（x, y, z）座標をÅ 単位で、またその広がりを$Å^2$ 単位で示しています。すべてのワニエ関数中心と広がりの総和を、「Total Center」で始まる行に示しています。

39.4 MLWF の作成例

様々な物質の MLWF の作成例として、ディレクトリ「work/wf_example」に入力ファイルが用意されています。

- Benzene.dat

  ベンゼンの 6 個の π 分子軌道から p_z 軌道型のワニエ関数を生成します。

- GaAs.dat

  GaAs の 4 つの価電子バンドから最局在ワニエ関数を生成します。

- Si.dat

  Si の価電子バンドおよび伝導バンドの両方を含む 8 つのワニエ関数を生成します。初期値は sp3 混成軌道です。

- symGra.dat

  グラフェンシートのワニエ関数を生成します。初期値は、炭素原子上の sp2 混成軌道と p_z 軌道です。

- pmCVO.dat

  非磁性の立方ペロブスカイト $CaVO_3$ に対する t_{2g} 型のワニエ関数を生成します。

- NC_CVO.dat

  「pmCVO.dat」の場合にスピン・軌道相互作用を含めた計算例です。

- GaAs_NC.dat

  「GaAs.dat」の場合にスピン・軌道相互作用を含めた計算例です。

- VBz.dat

  バナジウム・ベンゼン無限鎖に対するワニエ関数を生成します。これは、参考文献 [58] で研究されたものです。

39.5 出力ファイル

MLWF の計算で生成される 4 つのファイルを以下に説明します。拡張子「.mmn」のファイルには重なり行列要素 $M_{mn}^{(\mathbf{k},\mathbf{b})}$ が保存されています。拡張子「.amn」のファイルには初期推定のための射影行列要素 $A_{mn}^{(\mathbf{k})}$ が保存されています。拡張子「.eigen」のファイルには各 **k** 点における固有エネルギーと固有状態が保存されています。拡張子「.HWR」のファイルにはサンプリングした **k** グリッドと共役な Wigner-Seitz スーパーセル内にある、一組の格子ベクトル上での MLWF 間のホッピング積分が保存されています。最適化計算を再スタートする際には、拡張子「.mmn」のファイルが読み込まれます。4 つのファイルのより詳しい情報を以下に説明します。

A. ファイル「.mmn」の形式

このファイルの構造は、Wannier90 [87] のファイル構造に厳密に従っています。このファイルの最初の行は、第 2 行の数値についての説明です。第 2 行の数は、左から右に、それぞれ外エネルギー窓に含まれるバンド数 （N_{win}）、**k** 点の数、**b** ベクトルの数、スピン成分の数です。次に続く各行は $M_{mn}^{(\mathbf{k},\mathbf{b})}$ のデータブロックで、ループ形式で書き出されています。最も外側のループはスピン成分のループ、次は **k** 点のループ、そして **b** ベクトルのループです。また最も内側のループは、それぞれバンド・インデックスの n および m です。各ブロックにおいて、最初の行は 5 つの数から構成されます。最初の 2 つの数は、現在の **k** 点インデックスおよび隣接する **k+b** のインデックスです。次の 3 つの数は、単位セル **k+b** 点の座標です。ブロックの 2 行目からは、各行列要素の実部と虚部です。各ブロックにおいては、$N_{win} \times N_{win}$ 個の複素数があります。例として、入力ファイル「Si.dat」を用いて生成された「.mmn」ファイルを以下に示します。

```
Mmn_zero(k,b). band_num, kpt_num, bvector num, spinsize
           10          512            8            1
    1  512    0    0    0
    0.571090282808   -0.819911068319
    0.000031357498   -0.000045367307
   -0.000149292597    0.000215591228
   -0.003821911756    0.005522040495
    0.028616452988    0.019804944108
    0.003677357735    0.002544970842
   -0.006610037555   -0.004574771451
   -0.000950861169   -0.000658076633
   -0.000000008855    0.000000005272
    ........
    .....
    ...
```

B. ファイル「.amn」の形式

このファイルの構造は、Wannier90 [87] のファイル構造に厳密に従っています。ファイルの第 1 行はファ

イル全体の記述です。第 2 行の 4 つの数は、それぞれ外エネルギー窓内のバンドの数（N_{win}）、k 点の数、生成する MLWF の数、スピン成分の数です。同様に、データブロックはループ形式で書き出されています。最も外側のループはスピン成分で、その次が **k** 点、生成する MLWF、バンドの数です。このファイルの最初の行に示されているように、各ブロックにおいて、最初の 3 つの整数は、それぞれバンド・インデックス、MLWF インデックス、および **k** 点インデックスです。その後に、行列要素の実部と虚部 (Hartree) が出力されています。例として、入力ファイル「Si.dat」を用いて生成された「.amn」ファイルを以下に示します。

```
Amn. Fist line BANDNUM, KPTNUM, WANNUM, spinsize. Next is m n k...
          10          512            8            1
    1    1    1    0.053943539299    0.000161703961
    2    1    1   -0.000525446164   -0.000000008885
    3    1    1    0.002498021589    0.000000084311
    ... ...
    ... ...
   10    1    1   -0.000000023582   -0.000000000069
    1    2    1    0.053943534952    0.000161703965
    2    2    1    0.033382665372    0.000000493665
    3    2    1   -0.051189536188   -0.000001480360
    ........
    .....
```

C. ファイル「.eigen」の形式

このファイルは、各 k 点での固有エネルギーと固有状態を保存しています。最初の行は、系のフェルミ準位です。二行目にはバンドの数が示されています。次のデータは、主に二つの部分から成ります。最初の部分は固有エネルギーで、次の部分は固有状態に対応しています。各部分において、最も外側のループはスピン成分です。次のループは k 点で、その後にバンド・インデックス（指数、添字）が続きます。固有状態については、基底系に関して一つ以上の内部ループがあります。例として、入力ファイル「Si.dat」により生成されたファイルを以下に示します。

```
Fermi level -0.112747
Number of bands 10
    1    1   -0.566228100179
    2    1   -0.122518136808
    3    1   -0.122518129040
    4    1   -0.122518115949
    5    1   -0.026598417854
    ... ...
WF kpt 1 (0.00000000,0.00000000,0.00000000)
1 1   0.4790338281  -0.0014359768
1 2   0.0440709749  -0.0001321095
1 3  -0.0000003333  -0.0000000000
    ........
    .....
```

D. ファイル「.HWR」の形式

このファイルには、中心 ($\mathbf{R} = 0$) の単位セル内の m 番目の MLWF である $|m, \mathbf{0}\rangle$ と、$\mathbf{R}$ の単位セル内の n 番目の MLWF である $|n, \mathbf{R}\rangle$ の間のホッピング積分が保存されています。次のルールに従い、行列要素 $\langle m, \mathbf{0}|\hat{H}|n, \mathbf{R}\rangle$ は、書き出されます。最初の行は説明のみです。2 行目と 3 行目には、MLWF の数と Wigner-Seitz のスーパーセル内の格子ベクトルの数がそれぞれ書かれています。5、6、7 行目には単位セルベクトルが書かれています。8 行目には、コリニア計算かノンコリニア計算、またスピン分極の取扱いに関して説明されています。9 行目はフェルミ準位 (Hartree) です。10 行目からデータブロックが始まり、ループ形式で書き出されています。最も外側のループはスピン成分です。次のループは $\mathbf{R}$ について、最後の 2 つは m と n のループです。各 $\mathbf{R}$ は各ブロックの第 1 行目に縮退数とともに記載されています。各行においては、m と n のインデックスが表示され、その後にホッピング積分の実部と虚部 (Hartree) が続きます。例としてい、入力ファイル「Si.dat」により生成されたファイルを以下に示します。

```
Real-space Hamiltonian in Wannier Gauge on Wigner-Seitz supercell.
Number of Wannier Function 8
Number of Wigner-Seitz supercell 617
Lattice vector (in Bohr)
   5.10000    0.00000    5.10000
   0.00000    5.10000    5.10000
   5.10000    5.10000    0.00000
collinear calculation spinsize 1
Fermi level -0.112747
R (   -6    2    2 )    4
   1     1     -0.000078903162   -0.000000003750
   1     2      0.000024237763   -0.000000000148
   1     3      0.000024237691   -0.000000000341
   1     4      0.000024238375    0.000000004117
   1     5      0.000072656918   -0.000000000196
   1     6     -0.000022470544   -0.000000000859
   1     7     -0.000022481557    0.000000000750
   1     8     -0.000022492706    0.000000000148
   2     1      0.000024238091    0.000000000049
   2     2     -0.000078901874   -0.000000000011
   2     3      0.000024234912   -0.000000000023
    ........
    .....
    ...
```

39.6 MLWF の自動実行テスト

MLWF 計算に関連する機能が適切にインストールされていることを確認するために、MLWF 計算の自動実行テストを、次のようにして行うことができます。

MPI 並列計算の場合

```
% mpirun -np 16 openmx -runtestWF
```

OpenMP/MPI 並列計算の場合

```
% mpirun -np 8 openmx -runtestWF -nt 2
```

このテスト計算において、OpenMX は 8 個のテストケースを実行し、計算結果を「work/wf_example」に保存されている参照結果と比較します。比較結果（スプレッド関数と Ω 関数それぞれにおける絶対差異）が、「work」ディレクトリ中のファイル「runtestWF.result」に保存されます。参照結果は、Xeon のクラスターマシンを用いて計算されたデータです。絶対差異が小数点以下 7 桁以内であれば、インストールが正常であると判断されます。

40　対角化のための数値的に厳密な低次スケーリング法

大規模計算のために数値的に厳密な低次スケーリング法がサポートされています [80]。この方法の計算量は、N を基底関数の数とすると、1、2、3 次元の系に対してそれぞれ $O(N(\log N)^2)$、$O(N^2)$、$O(N^{7/3})$ として増加します。$O(N)$ 法と異なり、本方法は低次のスケーリングにも関らず通常の $O(N^3)$ 対角化法と同じく数値的に厳密な方法であり、一つの理論的な枠組で絶縁体から金属にも適用可能です。計算のデータ構造が適切に区分けされているため、図 36 に示すように大規模な並列処理に適しています。しかし計算量の前因子が大きくなるため、並列計算において多数の CPU コアを使用した場合にのみに、この方法が有利になります。多数の CPU コアを使って低次元の大規模系を計算する場合、この方法は適切な選択となるでしょう。この方法を選択するには、キーワード「scf.EigenvalueSolver」を以下の様に指定します。

```
scf.EigenvalueSolver     cluster2
```

本手法は、クラスター計算あるいはブリルアンゾーンでのサンプリングに対して Γ 点だけが考慮された周期系のコリニア DFT 計算に対してのみサポートされています。全エネルギー計算ばかりでなく、力の計算も実装されていますので、幾何学構造の最適化を行うことができます。しかし状態密度計算や波動関数の計算は実装されていません。周回積分の極の数 [55] は次のキーワードで設定します。

```
scf.Npoles.ON2              90
```

収束に必要な極の数は系の大きさに依らずに、系のスペクトル半径に依存しています [80]。電子温度が 300K 以上の場合には、100 の極を用いれば全エネルギーと力は十分に収束します。例として、「work」ディレクトリ中の入力ファイル「C60_LO.dat」を用いた計算を示します。

```
% mpirun -np 8 openmx C60_LO.dat
```

表 7 に示すように、本法で計算された全エネルギーは、倍精度の範囲内で通常の対角化法とほぼ一致していますが、一方、計算時間に関しては、従来法と比較し、非常に長い時間を要しています。計算時間に関し、本法と通常の対角化法の交点は、系の次元にも依りますが並列処理に 100 個以上のコアを使う場合、原子数で 300 程度であると推測されます。

表 7: 数値的に厳密な低次スケーリング法と通常の対角法の計算時間 (秒)。どちらの場合も、8 MPI プロセスを用いて C_{60} 分子の全エネルギーを計算。入力ファイルは「work」ディレクトリの 「C60_LO.dat」。

Method	Total energy (Hartree)	Computational time (sec.)
Low-order	-343.896238929370	69.759
Conventional	-343.896238929326	2.784

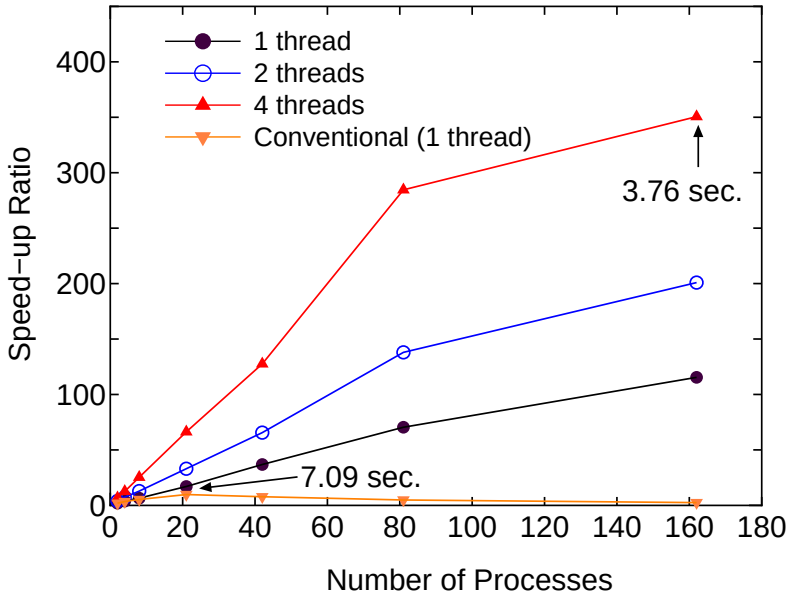


図 36: DNA の対角化 (SCF 計算) の OpenMP/MPI ハイブリッド並列計算の速度向上比。対角化には数値的に厳密な低次スケーリング法を使用。速度向上比は $2T_2/T_p$ として定義。T_2 は 2 MPI プロセスによる所要時間、T_p は並列計算の所要時間。これらの並列計算は AMD の Opteron クアッドコア・プロセッサ (2.3 GHz) からなる CRAY-XT5 マシン上で実施。電子温度を 700K とし、周回積分のための極の数は 80。比較のために、Householder 法と QR 法による通常の対角化法の並列計算の速度向上比も示す (シングルスレッドの場合)。また低次スケーリング法および通常法に対して、矢印で示した点での所要時間も示す。

41　有効遮蔽媒質法（ESM 法）

41.1　概要

有効遮蔽媒質法 (ESM 法) は、電荷を有するスラブモデルあるいは電場印加条件下にあるスラブモデルについての第一原理計算法です [81, 82, 83, 84]。この方法では、スラブの表面並行方向には 2 次元の周期境界条件を保ちつつ，垂直方向には 1 次元の境界条件を与え (図 37(a))、その 1 次元境界条件の下での Poisson 方程式をグリーン関数法によって解きます。実際には次のような半無限媒質を境界条件として導入することによって、孤立スラブ系、荷電スラブ系、および一様電場下にあるスラブ系を扱います。

(a) 孤立スラブ系：真空（比誘電率 $\varepsilon = 1$）+真空

(b) 荷電スラブ系：真空+理想金属（比誘電率 $\varepsilon = \infty$）

(c) 一様電場下スラブ：理想金属+理想金属

ここで、「スラブ」とは、通常の表面モデルとして使われるスラブばかりでなく、二次元的に配置された分子群から構成される系も含まれます。孤立スラブモデルは分極した系の解析に、また荷電スラブモデルは電極表面のシミュレーションに適用できます。二枚の理想金属媒質の間に挟まれたままで電場が印加されたスラブモデルは、金属コンデンサ内にある物質のモデル化に該当します。OpenMX において、ESM 法の計算で使用する単位セルは次のように構築します（図 37(a) 参照）。

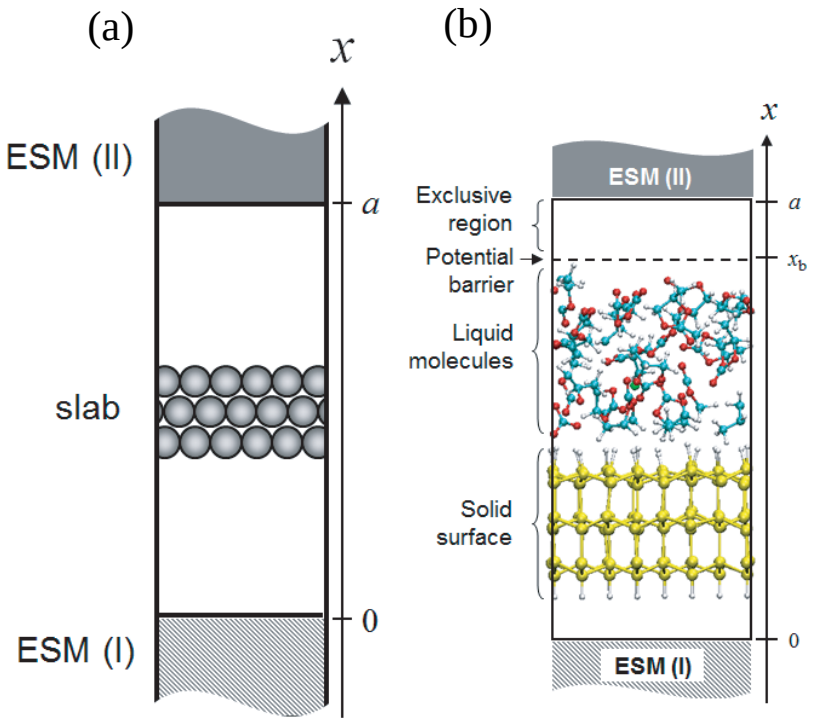


図 37: (a) 半無限媒質 (ESM) を伴うスラブの概略図。ESM(I) と ESM(II) はそれぞれ、x=0 と x=a（a は x 軸方向のセルの長さ）におけるセル境界。(b) ESM 法を用いた固液界面モデル系の MD 計算の単位セルの例。スラブと ESM は yz 面に平行に配置。

1. セルの a 軸は **b-c** 平面に対して垂直で x 軸に対して平行
2. 2 つの周期境界条件は y および z 軸方向に設定
3. ESM はセル境界（x=0, a）に配置
4. x 軸の原点はセル境界に設定
5. x 軸の分率座標は、0 と 1 の間で指定

ESM 法に基づく計算は次のキーワードによって実行します。

```
ESM.switch              on3      # off, on1=v|v|v, on2=m|v|m, on3=v|v|m, on4=on2+EF
ESM.buffer.range        4.5      # default=10.0 (ang),
```

ここで、on1、on2、on3、on4 は ESM の組み合わせ、すなわちそれぞれ「真空+真空」、「理想金属+理想金属」、「真空+理想金属」、「理想金属+電場中の理想金属」を表しています。キーワード「ESM.buffer.range」はモデル内原子に対する排他的領域の幅（Å 単位）を指定しており、これは波動関数と ESM の重なりを回避するために指定します。

1. ESM.switch = on1:

   ESM(I) および ESM(II) は半無限真空です。この条件は、計算する系の全電荷が中性である場合のみに適用されます。従って、キーワード「scf.system.charge」は必ずゼロに設定します。

2. ESM.switch = on2:

   ESM(I) および ESM(II) は半無限の理想金属です。電荷を持つ系を取り扱うことができます。キーワード「scf.system.charge」は有限な値に設定します。

3. ESM.switch = on3:

   ESM(I) および ESM(II) は、それぞれ、半有限の真空および理想金属です。電荷を持った系を取り扱うことができます。キーワード「scf.system.charge」は有限な値に設定します。

4. ESM.switch = on4:

   ここでは on2 と同じ ESM の組み合わせが適用されており、系に電場が印加されます。次のキーワードを使うことで計算する系に一様電場を印加することができます。

   ```
   ESM.potential.diff        1.0      # default=0.0 (eV),
   ```

   ここで、下部の理想金属を基準に 2 枚の半無限理想金属媒質間の電位差（eV 単位）を入力します。電場はセル長 a と電位差によって決まります。

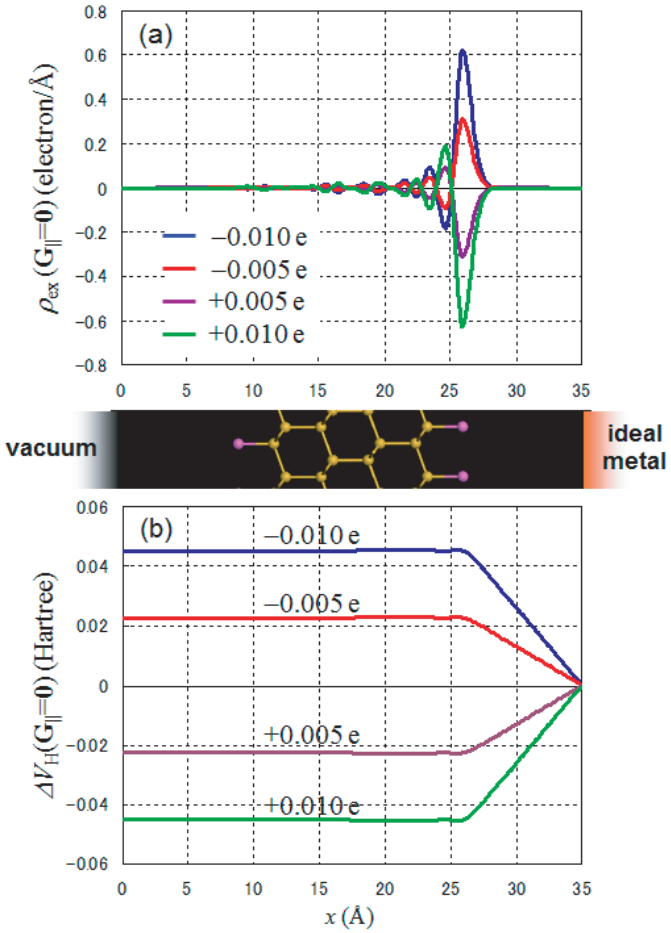


図 38: 真空+理想金属の条件における Si(111) スラブモデル。(a) Si(111) スラブ中の過剰電荷 ρ_{ex} の分布。(b) Si(111) スラブのバイアスによる Hartree ポテンシャルの変化 ΔV_H。ドープした電荷の量は-0.01、-0.005、+0.005、+0.01 e。各プロットは同じ ESM による中性のスラブを基準とした電荷または Hartree ポテンシャルの差の変化。

5. ESM 法を使った MD 計算

ESM 法を適用しながら固液界面系の MD 計算を行うことが出来ます。表面モデルスラブと液体領域は図 37(b) のように配置します。与えられた領域内に分子を制限するため、次のキーワードを使って障壁ポテンシャルを導入します（図 37(b) 参照)。

```
ESM.wall.position        6.0      # default=10.0 (ang)
ESM.wall.height        100.0      # default=100.0 (eV),
```

ここで、「ESM.wall.position」はセル上端と障壁ポテンシャルの原点の間の距離 $a - x_b$、また「ESM.wall.height」は $x = x_b + 1.0$ (Å) でのポテンシャルの高さ（ポテンシャルエネルギーの値）です。尚、MD を実行する間に表面モデルスラブの移動を防ぐために、そのスラブの最下層原子の座標を固定することをお勧めします。

41.2 テスト計算例

デモ計算として、図 38(a) に「真空+理想金属」（ESM.switch=on3）の境界条件の下で、1 × 1 のアルミニウム終端 Si(111) スラブ中の余剰電荷 ρ_{ex} 分布を示します（このテスト計算の入力ファイル「Al_Si111_ESM.dat」は work ディレクトリにあります)。ドープ電荷とそれに対応して発生する鏡像電荷との間の引力的相互作用により、スラブ内にあるドープ電荷が理想金属側の表面に偏析している様子がわかります。図 38(b) は、図 38(a) に示した各条件での Hartree ポテンシャル変化 ΔV_{H} を示しています。ここで、Al-Si(111) スラブ内部のポテンシャルと、スラブと理想金属間の電場はドープした電荷の量に従って変化します。

42 NEB (Nudged elastic band) 法

42.1 概要

OpenMX Ver. 3.7 では、幾何学的位相空間内の二つの安定構造をつなぐ最小エネルギー経路 (MEP) を調べるために、参考文献 [85] に基づく NEB 法がサポートされています。実装の詳細を以下にまとめます。

- 参考文献 [85] の (8)-(11) 式に基づく正接の計算
- 参考文献 [85] の (4) 式に基づく垂直力の計算
- 参考文献 [85] の (12) 式に基づく平行力の計算
- DIIS 法+BFGS 法によるハイブリッド最適化法

NEB の機能は構造最適化とほぼ同様な手続で利用可能です。そのため、比較的容易に本機能を使いこなすことができるでしょう。その特徴を以下に列挙します。

- 構造最適化とほぼ同様な手続で使用可能
- ハイブリッド OpenMP/MPI 並列化
- 直線初期経路またはユーザー定義による初期経路
- 3 つのルーチンのみを付加

42.2 実行方法

NBF の計算は次の三つのステップで実行されます。

1. 反応物の構造最適化
2. 生成物の構造最適化
3. 反応物と生成物をつなぐエネルギー経路の最適化

数値誤差の相殺するように、3 つの計算において同一の計算パラメータを使用して下さい。ここで計算パラメータとは単位格子セル、カットオフエネルギー、基底関数、擬ポテンシャル、電子温度等のことです。ステップ 1 およびステップ 2 の計算の後、「*.dat#」ファイルが作成されます。ファイル「*.dat#」中に記載されている原子座標を利用して、容易にステップ 3 に必要な入力ファイルを作成できます。ステップ 3 に対する入力ファイルが作成できれば、NEB の計算は、次のように通常の OpenMX 計算の場合と同様に実行できます。

```
% mpirun -np 32 openmx input.dat -nt 4
```

42.3 計算例と関連キーワード

計算例として二つの入力ファイルが用意されています。

- C2H4_NEB.dat

  2 個のエチレン分子の環状付加反応によるシクロブタンの生成

- Si8_NEB.dat

  ダイヤモンド構造の Si 中での格子間水素原子の拡散

以下では入力ファイル「C2H4_NEB.dat」を用いた NEB 計算を例に説明します。

反応物と生成物の構造を設定

反応物の原子座標を次のように入力ファイル中で指定します。

```
<Atoms.SpeciesAndCoordinates
   1    C   -0.66829065594143    0.00000000101783   -2.19961193219289     2.0     2.0
   2    C    0.66817412917689   -0.00000000316062   -2.19961215251205     2.0     2.0
   3    H    1.24159214112072   -0.92942544650857   -2.19953308980064     0.5     0.5
   4    H    1.24159212192367    0.92942544733979   -2.19953308820323     0.5     0.5
   5    H   -1.24165800644131   -0.92944748269232   -2.19953309891389     0.5     0.5
   6    H   -1.24165801380425    0.92944749402510   -2.19953309747076     0.5     0.5
   7    C   -0.66829065113509    0.00000000341499    2.19961191775648     2.0     2.0
   8    C    0.66817411530651   -0.00000000006073    2.19961215383949     2.0     2.0
   9    H    1.24159211310925   -0.92942539308841    2.19953308889301     0.5     0.5
  10    H    1.24159212332935    0.92942539212392    2.19953308816332     0.5     0.5
  11    H   -1.24165799549343   -0.92944744948986    2.19953310195071     0.5     0.5
  12    H   -1.24165801426648    0.92944744880542    2.19953310162389     0.5     0.5
Atoms.SpeciesAndCoordinates>
```

生成物の原子座標を次のように入力ファイル中で指定します。

```
<NEB.Atoms.SpeciesAndCoordinates
   1    C   -0.77755846408657   -0.00000003553856   -0.77730141035137     2.0     2.0
   2    C    0.77681707294741   -0.00000002413166   -0.77729608216595     2.0     2.0
   3    H    1.23451821718817   -0.88763832172374   -1.23464057728123     0.5     0.5
   4    H    1.23451823170776    0.88763828275851   -1.23464059022330     0.5     0.5
   5    H   -1.23506432458023   -0.88767426830774   -1.23470899088096     0.5     0.5
   6    H   -1.23506425800395    0.88767424658723   -1.23470896874564     0.5     0.5
   7    C   -0.77755854665393    0.00000000908006    0.77730136931056     2.0     2.0
   8    C    0.77681705017323   -0.00000000970885    0.77729611199476     2.0     2.0
   9    H    1.23451826851556   -0.88763828740000    1.23464060936812     0.5     0.5
  10    H    1.23451821324627    0.88763830875131    1.23464061208483     0.5     0.5
  11    H   -1.23506431230451   -0.88767430754577    1.23470894717613     0.5     0.5
  12    H   -1.23506433587007    0.88767428525317    1.23470902573029     0.5     0.5
NEB.Atoms.SpeciesAndCoordinates>
```

NEB 計算のためのキーワード

NEB 計算は次のようにキーワード「MD.Type」を設定することによって実行します。

```
MD.Type                 NEB
```

経路中のイメージ数を以下のキーワードで指定します。

```
MD.NEB.Number.Images        8           # default=10
```

ここで、2 つの終端構造 (反応物と生成物) はイメージの数から除きます。
バネ定数は次のように与えます。

```
MD.NEB.Spring.Const         0.1         # default=0.1(hartee/borh^2)
```

ほとんどの場合、得られる経路はこの値に大きく依存しません。
MEP の最適化はハイブリッド最適化法 (DIIS+BFGS) によって行います。この最適化法は次のキーワードによって制御されます。

```
MD.Opt.DIIS.History          4          # default=7
MD.Opt.StartDIIS            10          # default=5
MD.maxIter                 100          # default=1
MD.Opt.criterion         1.0e-4         # default=1.0e-4 (Hartree/Bohr)
```

これらのキーワードの仕様は、構造最適化に対するものと同様なので、詳細についてはマニュアルの「構造最適化」の章をご覧下さい。またキーワード「MD.Fixed.XYZ」で原子の位置を固定することも可能です。

NEB 計算の実行

以下の様に、入力ファイル「C2H4_NEB.dat」を用いて NEB 計算を実行できます。

```
% mpirun np 16 openmx C2H4_NEB.dat
```

計算が正常に完了すると、24 個以上のファイルが作成されます。これらのファイルには次のようなものがあります。

```
c2h4.neb.opt          history of optimization for finding MEP
c2h4.neb.ene          total energy of each image
c2h4.neb.xyz          atomic coordinates of each image in XYZ format
C2H4_NEB.dat#         input file for restarting.
C2H4_NBE.dat_0        input file for the precursor
C2H4_NBE.dat_1        input file for the image 1
C2H4_NBE.dat_2        input file for the image 2
C2H4_NBE.dat_3        input file for the image 3
C2H4_NBE.dat_4        input file for the image 4
C2H4_NBE.dat_5        input file for the image 5
C2H4_NBE.dat_6        input file for the image 6
```

```
C2H4_NBE.dat_7        input file for the image 7
C2H4_NBE.dat_8        input file for the image 8
C2H4_NBE.dat_9        input file for the product
c2h4_0.out            output file for the precursor
c2h4_1.out            output file for the image 1
c2h4_2.out            output file for the image 2
c2h4_3.out            output file for the image 3
c2h4_4.out            output file for the image 4
c2h4_5.out            output file for the image 5
c2h4_6.out            output file for the image 6
c2h4_7.out            output file for the image 7
c2h4_8.out            output file for the image 8
c2h4_9.out            output file for the product
```

「c2h4.neb.opt」には、図 39(a) に示すように、MEP の最適化過程の履歴が保存されています。以下に「c2h4.neb.opt」を示すように、ファイルのヘッダーに説明が記載されています。

```
***********************************************************
***********************************************************
         History of optimization by the NEB method
***********************************************************
***********************************************************
```

iter	SD_scaling	|Maximum force| (Hartree/Bohr)	Maximum step (Ang)	Norm (Hartree/Bohr)	Sum of Total Energy of Images (Hartree)
1	0.37794520	0.12552253	0.04583483	0.49511548	-223.77375997
2	0.37794520	0.08735938	0.03172307	0.35373414	-223.85373393
3	0.37794520	0.05559291	0.01919790	0.25650527	-223.89469352
4	0.37794520	0.03970051	0.01254863	0.20236344	-223.91689564
5	0.45353424	0.03132536	0.01360864	0.17275416	-223.93128189
6	0.45353424	0.02661456	0.01202789	0.15142709	-223.94412534
7	0.45353424	0.02367627	0.01068250	0.13703973	-223.95422398
.....					
...					
.					

また「c2h4.neb.ene」と「c2h4.neb.xyz」は、図 39(b) に示すように反応物からの距離（Bohr）の関数としての全エネルギーの変化および構造の変化が保存されていますので、MEP の分析に利用できます。「c2h4.neb.ene」の内容は次のようなものです。

```
#
# 1st column: index of images, where 0 and MD.NEB.Number.Images+1 are the terminals
# 2nd column: Total energy (Hartree) of each image
# 3rd column: distance (Bohr) between neighbors
# 4th column: distance (Bohr) from the image of the index 0
```

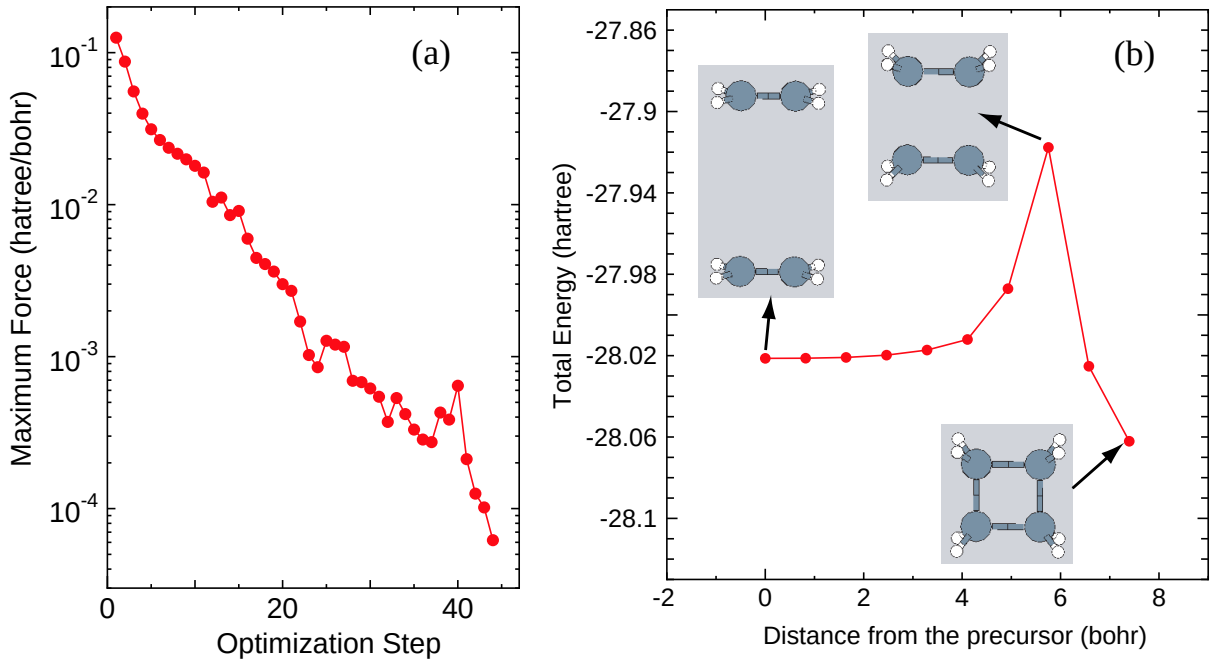


図 39: (a) 二個のエチレン分子の環状付加反応 (シクロブタンの生成) の NEB 計算の最適化履歴 (c2h4.neb.opt)、(b) 反応物からの距離（Bohr）の関数としての 2 つのエチレン分子の全エネルギーの変化 (c2h4.neb.ene) と、それに対応する最小エネルギー経路上のイメージの幾何構造 (c2h4.neb.xyz)。本 NEB 計算の入力ファイルは「work」ディレクトリ内の「C2H4_NEB.dat」。

```
#
    0      -28.02131967      0.00000000      0.00000000
    1      -28.02125585      0.82026029      0.82026029
    2      -28.02086757      0.82124457      1.64150486
    3      -28.01974890      0.82247307      2.46397794
    4      -28.01724274      0.82231749      3.28629543
    5      -28.01205847      0.82220545      4.10850088
    6      -27.98707448      0.82271212      4.93121300
    7      -27.91765377      0.82175187      5.75296486
    8      -28.02520689      0.82164937      6.57461423
    9      -28.06207901      0.82095145      7.39556568
```

ここで第 1 列はイメージの通し番号で、0 と 9 はそれぞれ反応物と生成物に相当します。第 2 列はそれぞれのイメージの全エネルギーです。第 3 および第 4 列は、幾何学的位相空間内の 2 つの隣接イメージの間の距離（Bohr）と反応物からの距離（Bohr）です。それぞれのイメージの計算は、基本的に異なる入力ファイルによる独立した OpenMX の計算として実行されるため、「*.dat_#」ファイルが自動生成されています。ここで「*」は「System.Name」、「#」はそれぞれのイメージの通し番号です。対応する出力ファイル「*_#.out」も出力されますが、これは MEP 上で電子構造がどのように変化するのか、分析するのに役立つでしょう。

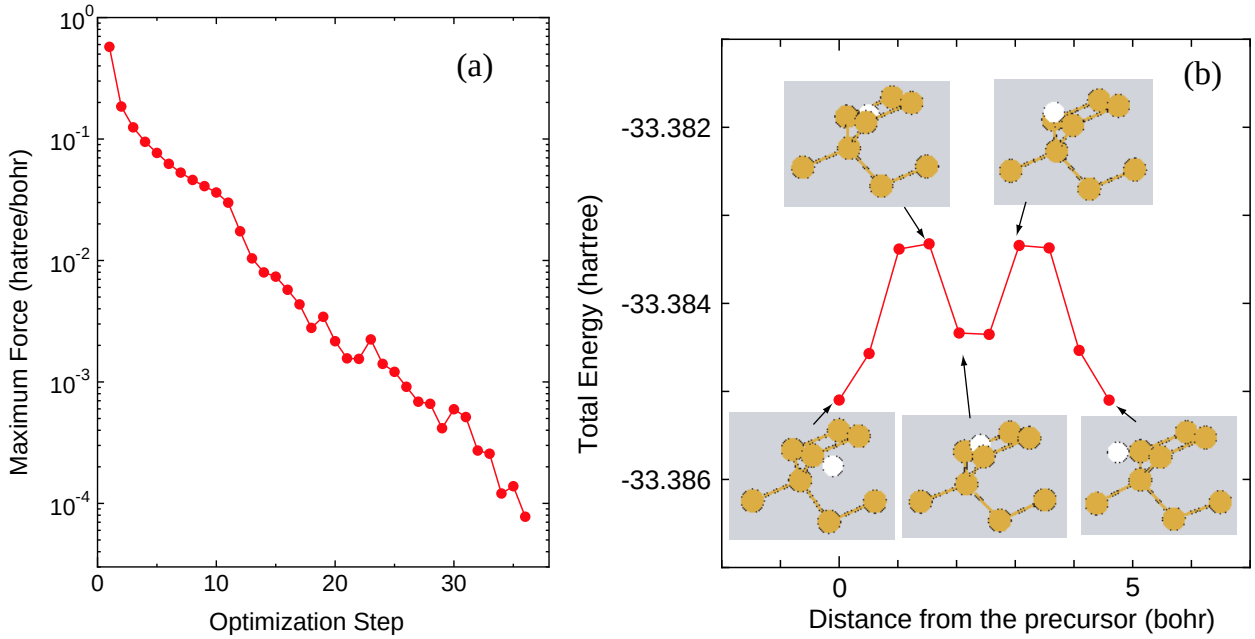


図 40: (a) ダイヤモンド Si 中の格子間水素原子の拡散に対する NEB 計算の最適化履歴 (si8_neb.neb.opt)、(b) 反応物からの距離（Bohr）の関数としての全エネルギーの変化 (si8_neb.neb.ene) と、それに対応する最小エネルギー経路上のイメージの幾何構造 (si8_neb.neb.xyz)。本 NEB 計算の入力ファイルは「work」ディレクトリ内の「Si8_NEB.dat」。

「C2H4_NEB.dat」と同様に、「Si8_NEB.dat」でも NEB の計算を行う事ができます。計算が正常に終了すれば、図 40 に示す最適化の履歴と MEP に沿った全エネルギー変化が得られるでしょう。

42.4 NEB 計算の再スタート

最適化ステップ数が最大値に達しても計算が収束しないことがしばしば発生します。この場合には、前のジョブの最後の最適化ステップから、新しいジョブとして最適化を続けて下さい。ファイル「*.dat#」が最適化ステップ毎に生成されています。このファイルには、最終ステップでの全てのイメージの原子座標が設定されています。この入力ファイル「*.dat#」を使って最適化計算を再スタートできます。

42.5 ユーザー定義の初期経路

デフォルトで、反応物と生成物をつなぐ初期経路は、それらを結ぶ直線です。しかし、系によっては直線上に生成されたイメージの幾何学構造は原子間距離が近接してしまい、物理的には許容できないものになります。この場合、初期イメージの原子座標を明示的に与える必要があります。ユーザー自身で初期イメージの原子座標を明示的に与える方法は、実は再スタートの場合と同様です。どちらの場合でも次のキーワードによって、それぞれのイメージの原子座標を明示的に与えます。

```
<NEB1.Atoms.SpeciesAndCoordinates
   1    Si    -0.12960866043083     0.13490502997627    -0.12924862991035      2.0      2.0
   2    Si    -0.40252421446808     5.19664433048606     4.91248322056082      2.0      2.0
   ...
```

```
NEB1.Atoms.SpeciesAndCoordinates>

<NEB2.Atoms.SpeciesAndCoordinates
   1   Si   -0.08436294149342   -0.02173837971883   -0.08374099211565     2.0     2.0
   2   Si   -0.33677725120015    5.10216241168093    5.01087499461541     2.0     2.0
   ...
NEB2.Atoms.SpeciesAndCoordinates>
```

「MD.NEB.Number.Images」で指定した全てのイメージに対して、原子座標を与える必要があります。さらに、キーワード「scf.restart」を「on」に設定して下さい。

```
scf.restart      on
```

42.6 NEB 計算における SCF の確認

NEB 計算での標準出力には、イメージ 1 の情報のみが表示されます。他のイメージについての情報は表示されません。SCF の反復計算が全てのイメージに対して収束するという保証はありませんので、ユーザー各自で SCF 計算の収束性を確認する必要があります。全てのイメージについて SCF 収束をモニタするために、一時ファイルが利用できます。NEB 計算では、それぞれのイメージに対して一つの入力ファイル「*.dat_#」が作成されます。ここで「#」は 0 から MD.NEB.Number.Images+1 までの値で、それぞれの入力ファイル中の「system.name」は「system.name_#」に変更されています。従って、ファイル「system.name_#.DFTSCF」をモニタすることによって各イメージの SCF 計算の収束状況を確認できます。

42.7 並列計算

NEB 計算の並列化の設定はプロセスとスレッドの数に応じて自動的に行われます。ただし MPI 並列化で適正な計算負荷を実現するためには、適切なプロセス数を設定する必要があります。ここで適切な MPI プロセス数とは、「MD.NEB.Number.Images」で与えられるイメージ数で割り切れる MPI プロセス数のことです。この場合には計算負荷が最適化されます。また原子数を上回る MPI 並列プロセス数を設定した場合においても、計算負荷を考慮し、適切な並列計算が実行されます。他の機能の並列化と同様に、OpenMP/MPI によるハイブリッド並列化もサポートしています。ほとんどの場合に、デフォルトの並列化の設定で問題なく動作しますが、大規模系の計算において少数の MPI プロセスを用いるとメモリー不足が重大な問題となり得ます。デフォルトの MPI 並列化では、イメージが優先的に並列化されます。MPI プロセス数がイメージ数を上回ると、次にそれぞれの各イメージ内の計算が並列化されます。その際に、メモリーもまた同様に並列化されます。そのため、デフォルトの並列化設定では、多数の MPI プロセスが利用できなければメモリー不足のために異常終了する可能性があります。この状況を避けるために次のキーワードが利用できます。

```
MD.NEB.Parallel.Number    3
```

この例では、イメージを 3 つのグループに分けてそれぞれで並列計算を行います。そのため、MPI プロセスを 3 つのグループに分類し、各 MPI グループを各イメージのグループの計算に割り当てます。並列化構造は 2 階層となっており、上の階層がイメージのグループの並列化、下の階層はそれぞれのイメージ内の計算の並列化です。全てのイメージの計算を完了するために、グループ化した計算を最低 [(イメージ数)/(MD.NEB.Parallel.Number)] 回繰り返します。「MD.NEB.Parallel.Number=1」と設定した場合には、イメージの並列化は行わずに、イメージ内の計算に全ての MPI プロセスが割り当てられることになり、この場合にメモリ使用量が最小になります。したがって、この並列化法は大規模系の NEB 計算に有効でしょう。このキーワードが入力ファイルで指定されない場合には、デフォルトの並列化法が適用されます。

42.8 その他の注意

バルク系に対しては「FRAC」の替わりに「Ang」または「AU」で原子座標を指定して下さい。「FRAC」で指定した場合には、原子の位置は 0 から 1 の間の規格化座標に自動的に変換された後に、それに対応するカルテシアン座標が生成されます。この場合、幾何空間中でイメージが連続的に配置されないことになり、NEB 計算が破綻します。これを避けるために、「Ang」または「AU」で原子座標を指定して下さい。

NEB の機能を実装するために、3 つのルーチン、「neb.c」、「neb_run.c」、「neb_check.c」のみが追加されています。メインルーチンは「neb.c」です。関連する方法を実装する場合には、ルーチン「neb.c」を参照して下さい。

43 Tersoff-Hamann 法による STM イメージ

Tersoff-Hamann 法 [52] によって走査型トンネル顕微鏡（STM）のイメージを計算することができます。この方法は、化学ポテンシャルから計ったエネルギーウィンド内の部分電荷密度の計算に他なりません。部分電荷密度は次のキーワードを使って計算します。

```
partial.charge                  on       # on|off, default=off
partial.charge.energy.window    0.0      # in eV
```

ここで、二番目のキーワードにより化学ポテンシャル（プラスの値は伝導帯を、マイナスは価電子帯を意味します）から計ったエネルギーウィンドウ（eV 単位）を指定します。部分電荷密度の計算は状態密度（DOS）の計算の際に実行されますので、次のキーワードも指定する必要があります。

```
Dos.fileout                    on        # on|off, default=off
Dos.Erange                -20.0  20.0    # default = -20 20
Dos.Kgrid                    5 5 5       # default = Kgrid1 Kgrid2 Kgrid3
```

計算が正常に終了すれば、Tersoff-Hamann 近似の範疇で STM シミュレーションに利用できる「*.pden.cube」が生成されます。例として、本手法によって得られたグラフェン層の STM 像を図 41 に示します。

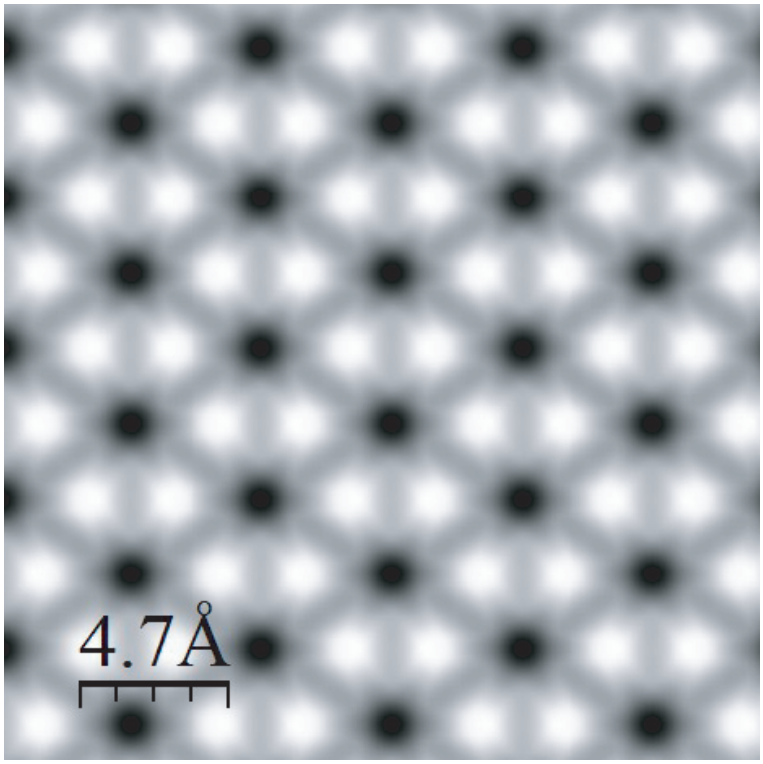


図 41: Tersoff-Hamann 近似によるグラフェン層の STM 像。計算では「partial.charge.energy.window」に 2eV に設定。入力ファイルはディレクトリ「work」中の「Graphene_STM.dat」。ソフトウェア WSxM [92] を使い、cube ファイル「Graphene_STM.pden.cube」を等値面値 0.0001 で可視化。

44　vdW 相互作用のための DFT-D2 法

Grimme による DFT-D2 法 [86] により、vdW 相互作用を組み込むことができます。関連するキーワードを以下に列挙します。

```
scf.dftD                    on          # on|off, default=off
DFTD.Unit                   Ang         # Ang|AU
DFTD.rcut_dftD              100.0       # default=100 (DFTD.Unit)
DFTD.d                      20.0        # default=20
DFTD.scale6                 0.75        # default=0.75
DFTD.IntDirection           1 1 1       # default=1 1 1 (1:on 0:off)
```

vdW 相互作用を組み込む際には、「scf.dftD」を「on」に設定して下さい。二体ポテンシャルのカットオフ半径はキーワード「DFTD.rcut_dftD'」で指定します。ここで、その単位は「DFTD.Unit」で与えます。Grimme の論文 [86] にある (12) 式の「d」値は「DFTD.d」によって指定します。そのデフォルト値は 20 です。Grimme の論文 [86] にある (11) 式の倍率は「DFTD.scale6」で指定します。PBE 関数を用いた場合のデフォルト値は 0.75 です。また本手法で導入された vdW 相互作用は「DFTD.IntDirection」によって **a**、**b**、もしくは **c** 軸方向で打ち切ることが可能で、ここで「1」は相互作用を含み、「0」は含まない場合です。また次のように、それぞれの原子の周期性を個別に制御できます。

```
<DFTD.periodicity
   1   1
   2   1
   3   1
   4   1
  ....
DFTD.periodicity>
```

ここで第 1 列は「Atoms.SpeciesAndCoordinates」と同様の通し番号、第 2 列はフラグで、対応する原子に対して「1」は周期的、「0」は非周期的であることを意味します。「0」を指定した原子は非周期的であると見なされ、周期セルとの相互作用は含まれません。

vdW 相互作用を実装する上での主な変更は二つのルーチン、「Total_Energy.c」中の「Calc_EdftD()」と「DFTDvdW_init.c」に成されています。「DFTDvdW_init.c」中では、vdW 補正のパラメータが容易に変更でき、「Total_Energy.c.」の「Calc_EdftD()」ではどのように実装されているのか確認できます。

OpenMX では基底関数として局在軌道を用いているため、vdW 相互作用のような弱い相互作用の効果を調べる際には、基底関数重なり誤差（BSSE）を考慮する事が重要です。BSSE を評価するために counterpoise 法（CP 法）[33, 34] が使用できます。CP 法については「空原子の配置」の節を参照して下さい。

45 “格子定数 vs. エネルギー” 曲線の計算

45.1 “格子定数 vs. エネルギー” 曲線

次のキーワードによって “格子定数 vs. エネルギー” 曲線を計算することができます。

```
MD.Type                    EvsLC      #
MD.EvsLC.Step               0.4       # default=0.4%
MD.EvsLC.flag             1  1  1     # a, b, c-軸に対して, default=1 1 1
MD.maxIter                  32        # default=1
```

「MD.Type」を「EvsLC」に設定すると、自動的に単位セルベクトル **a**、**b**、**c** を逐次的に変化させることで、簡単に “格子定数 vs. エネルギー” 曲線が計算できます。一定比率で単位セルベクトルを変化させながら、全エネルギーの計算が実施されます。セルベクトルの変化量は「MD.EvsLC.Step」で与えられます。ここで単位はパーセントです。また初期単位セルベクトルが基準値となります。デフォルトの設定では、単位セルベクトル **a**、**b**、**c** を一様に変化させて計算を行いますが、「MD.EvsLC.flag」によって、変化させる軸を選択することも可能です。「1」は格子定数を変化,「0」は格子定数を変化させない指定となります。計算のステップ数はキーワード「MD.maxIter」で指定します。計算が正常に終了すると、ファイル「*.EvsLC」が生成されます (ファイル「*.EvsLC」は計算途上でも、途中までの計算結果が出力されています)。ここで「*」は「System.Name」です。「*.EvsLC」中の列は、Å 単位での a_x、a_y、a_z、b_x、b_y、b_z、c_x、c_y、c_z および全エネルギー (Hartree 単位) の順に配列されています。ここで、$a(b,c)_x$、$a(b,c)_y$、$a(b,c)_z$ はベクトル **a**(**b**,**c**) の x、y、z 成分です。例として、fcc 構造の Mn 固体の” 格子定数 vs. エネルギー” 曲線を図 42 に示しました。

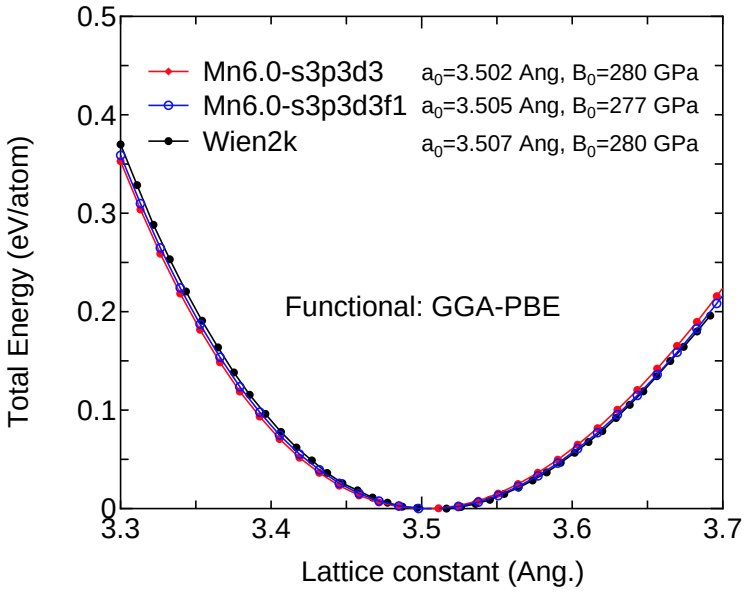


図 42: キーワード「EvsLC」によって計算された fcc 構造の Mn 固体の “格子定数 vs. エネルギー” 曲線。使用した入力ファイルはディレクトリ「work」中の「Mnfcc-EvsLC.dat」。図中の平衡格子定数と体積弾性率は、ウェブサイト [88] で提供されている「murn.f」コードを用い、Murnaghan 状態方程式にデータをフィッティングすることによって評価。

45.2 デルタ因子

前節の「EvsLC」の他に、類似する機能が「DF」で利用可能です。

```
MD.Type                         DF
```

このキーワードにより、OpenMX は自動的に、入力ファイルで与えられる初期構造を基準として、-6、-4、-2、0、2、4、6%の体積を持つ系の全エネルギーを計算します。体積の調節は、単純に格子ベクトル **a**、**b**、**c** 軸を一様に変化させることで行います。体積とそれに対応する全エネルギーはファイル「*.DF」に出力されます。ここで「*」は「System.Name」です。このデータは文献 [27] で提案されているデルタ因子の計算に利用できます。

46　フェルミ面

フェルミ面は XCrySDen [61] を使って可視化できます。フェルミ面の描画に必要なデータは状態密度の計算の際に生成されます。状態密度の計算は次のキーワードを用いて実行します (詳細は「状態密度」の章を参照して下さい)。

```
Dos.fileout                 on          # on|off, default=off
Dos.Erange               -20.0   20.0   # default = -20 20
Dos.Kgrid                  61 61 61     # default = Kgrid1 Kgrid2 Kgrid3
```

計算が正常に終了すると、「*.FermiSurf0.bxsf」(「*」は「System.Name」) というファイルが生成されます。このファイルは「XCrySDen」[61] を用いて可視化できます。「Dos.Fileout」と同様に、この目的にために「DosGauss.fileout」も使用可能です。スピン分極計算の場合には、アップスピンとダウンスピンに対して、それぞれ「*.FermiSurf0.bxs」と「*.FermiSurf1.bxs」と名付けられた 2 つのファイルが生成されます。ノンコリニア計算の場合、「*.FermiSurf.bxs 」というファイルが生成されます。滑かなフェルミ面を描画するには、多数の **k** 点が必要になることに注意して下さい。例として、fcc 構造の Ca 固体のフェルミ面を図 43 に示します。この計算に使用された入力ファイルは、ディレクトリ「work」中の「Cafcc_FS.dat」です。

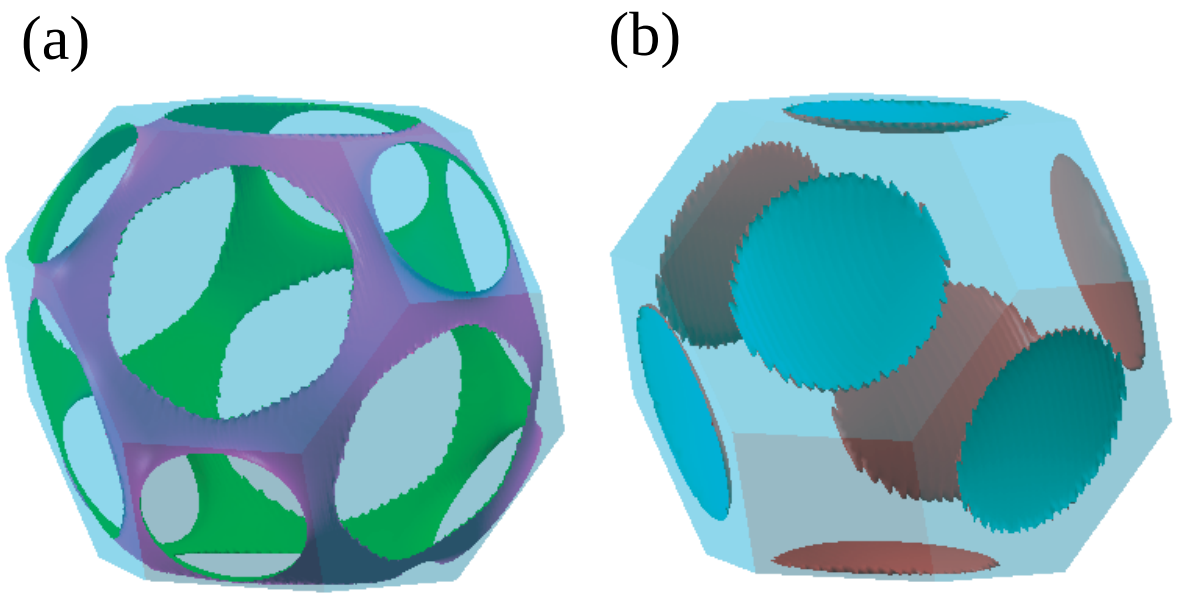


図 43:「 XCrySDen」[61] で可視化した fcc 構造の Ca 固体のフェルミ面。2 種類のバンドがフェルミエネルギー (化学ポテンシャル) を横切るため、2 種類のフェルミ面を (a) と (b) に表示。計算に使用された入力ファイルは、ディレクトリ「work」中の「Cafcc_FS.dat」。

47　2つの Gaussian Cube ファイルの差の解析

ユーティリティツールによって、2 つの Gaussian cube ファイルの差の解析を行うことができます。構造変化やスピン磁気モーメント方位の変化に伴い、どのように全電子密度、スピン密度、ポテンシャルが変化しているのか解析することが可能です。

(1) diff_gcube.c のコンパイル

ディレクトリ「source」内に、「diff_gcube.c」という名のファイルが保存されています。このファイルを次のコマンドでコンパイルします。

```
% gcc diff_gcube.c -lm -o diff_gcube
```

コンパイルが正常に終了すると、ディレクトリ「source」内に、実行可能なファイル「diff_gcube」が生成されます。この実行可能ファイルをディレクトリ「work」にコピーして下さい。

(2) 差の計算

「input1.cube」と「input2.cube」という名前の 2 つの Gaussian cube ファイルの差を求め、さらにその結果をファイル「output.cube」に出力したい場合、「diff_gcube」を次のように実行します。

```
% ./diff_gcube input1.cube input2.cube output.cube
```

差は Gaussian cube 形式で「output.cube」という名前のファイルに出力されます。「output.cube」は、XCrySDen [61] や Molekel [60] 等の可視化ソフトウエアを用いて、容易に解析できます。実際に、「電場」の章の図 22 はこの方法で作成されました。

48　2つの幾何構造の差の解析

ユーティリティツールによって、3次元座標（デカルト座標）を記録している2つの xyz ファイルの差を解析することができます。電場や置換基の影響により、どのように構造が変化しているのか解析することが可能です。幾何構造の差の解析には以下で説明する3種類の解析方法が利用できます。

2つの幾何構造間の標準偏差（RMSD）は次式で定義されます。

$$\mathrm{RMSD} = \sqrt{\frac{\sum_i^{N_{\mathrm{atom}}} (R_i - R_i^0)^2}{N_{\mathrm{atom}}}}.$$

2つの幾何構造間の平均偏差（MD）は次式で定義されます。

$$\mathrm{MD} = \frac{\sum_i^{N_{\mathrm{atom}}} |R_i - R_i^0|}{N_{\mathrm{atom}}}.$$

2つの幾何構造間の結合距離の平均偏差（MDBL）は次式で定義されます。

$$\mathrm{MDBL} = \frac{\sum_i^{N_{\mathrm{bond}}} |BL_i - BL_i^0|}{N_{\mathrm{bond}}}.$$

ここで、N_{atom} と N_{bond} は、それぞれ原子数、カットオフ半径内の結合長（BL）を持つ結合数です。また、それぞれの原子の xyz 座標間の偏差ベクトルは、XCrySDen 形式で「dgeo_vec.xsf」という名前のxsf ファイルに出力されますので、2つの構造間の差を可視化できます。

以下に解析の手順を説明します。

(1) diff_gcube.c のコンパイル

ディレクトリ「source」内にファイル「diff_geo.c」があります。このファイルを次のコマンドでコンパイルします。

```
% gcc diff_geo.c -lm -o diff_geo
```

コンパイルが正常に終了すると、ディレクトリ「source」内に実行ファイル「diff_geo.c」が生成します。この実行ファイルをディレクトリ「work」にコピーして下さい。

(2) 2つの幾何構造間の差の計算

diff_geo.c のヘッダー部分に次のような「使用法 (usage)」が出ています。

```
usage:
        ./diff_geo file1.xyz file2.xyz -d rmsd


   option
      -d rmsd      a root mean square of deviation
      -d md        a mean deviation
      -d mdbl 2.2  a mean deviation between bond lengths,
                   2.2 (Ang) means a cutoff bond length which
                   can be taken into account in the calculation
```

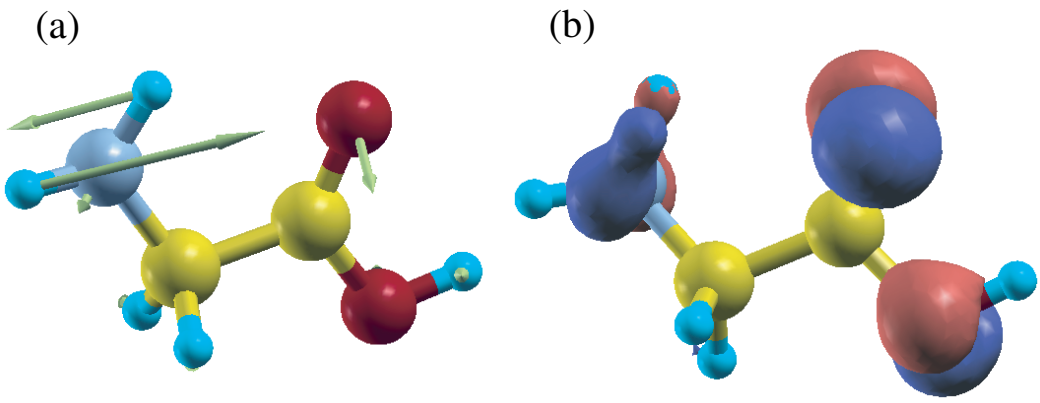


図 44: (a) 最適化構造間の原子座標の差に対応するベクトル。中性グリシン分子と電子を 1 個付加したグリシン分子の構造最適化をそれぞれ行い、最適化構造間の比較を実施。(b) 2 つの系の全電子密度の差。青色と赤色は、それぞれ全電荷密度の減少と増加を示す。図は XCrySDen を使って可視化。

2 つの幾何構造間の RMSD を知りたければ、次のように実行します。

```
% ./diff_geo file1.xyz file2.xyz -d rmsd
```

計算結果は標準出力に表示されます。また XCrySDen のベクトル形式で、各原子の 3 次元座標の差がファイル「dgeo_vec.xsf」に保存されます。このファイルは XCrySDen 中の「Display→Forces」を使用して、可視化できます。MDBL を計算する場合には、カットオフ結合距離（Å）を与えて下さい。RMSD 計算ではカットオフ半径内にある結合距離が考慮されます。図 44 (a) に最適化構造間の原子座標の差に対応するベクトルを示します。この計算では中性グリシン分子と電子を 1 個付加したグリシン分子の構造最適化をそれぞれ行い、最適化構造間の比較を行いました。また図 44 (b) に 2 つの系の全電子密度の差を示します。大きな電荷の変化に伴って、大きな構造変化が生じていることが分かります。この例から、電荷のドーピングや静電場によって構造変化がどのように起こるのかを知るために、このツールが有用であることが分かります。

49　相互作用によって引き起こされる電荷密度の差の解析

２つの系 A と B の間の相互作用によって引き起こされる（スピン）電子密度の再配分は、次の手順によって解析できます。

(i) A と B から成る複合系の計算

A と B から成る複合系に対して、通常の計算を行い、（スピン）電子密度のための cube ファイルを生成します。このファイルを「AB.cube」とします。また標準出力から「Grid_Origin」の情報を取得します。これは一様グリッドを生成する際の原点の x、y、z 成分となります。

```
Grid_Origin  xxx  yyy  zzz
```

この値は次の (ii) と (iii) で議論する計算において使用します。

(ii) 系 A の計算

A のみを含んだ系の通常の計算を行います。ただし (i) の計算と同一の単位胞を使用して下さい。また計算条件も同一に設定する必要があります。また系 A の幾何構造は、計算 (i) の場合と同一でなければなりません。一様グリッドを生成する際の原点の x、y、z 成分も (i) と同一に設定するために、入力ファイル中に次のキーワードを加えて下さい。

```
scf.fixed.grid  xxx  yyy  zzz
```

ここで、「xxx yyy zzz」は (i) の「Grid_Origin」から取得した座標です。このようにして、系 A に対する cube ファイルが得られます。これを「A.cube」とします。

(iii) 系 B の計算

(ii) の計算と同様にして、系 B のみの計算も A と B から成る複合系の場合と同じ単位胞を使用し、また同一の計算条件で実行する必要があります。また、系 B の幾何構造は (i) の場合と同一でなければなりません。一様グリッドを生成する際の原点の x、y、z 成分も (i) と同一に設定するために、入力ファイル中に次のキーワードを加えて下さい。

```
scf.fixed.grid  xxx  yyy  zzz
```

ここで、「xxx yyy zzz」は (i) の「Grid_Origin」から取得した座標です。このようにして、系 B に対する cube ファイルが得られます。これを「B.cube」とします。

(iv) 2 つのコードをコンパイルする

2 つのコードを次のようにしてコンパイルします。

```
% gcc diff_gcube.c -lm -o diff_gcube
% gcc add_gcube.c -lm -o add_gcube
```

(v) （スピン）電子密度の差を解析するための cube ファイルの作成

まず、次のコマンドで、系 A と B の 2 つの（スピン）電子密度の重ねあわせの cube ファイルを作成します。

```
% ./add_gcube A.cube B.cube A_B.cube
```

ファイル「A_B.cube」は 2 個の孤立系の（スピン）電子密度の重ねあわせの cube ファイルです。次に、相互作用によって引き起こされる（スピン）電子密度の差の cube ファイルを、次のようにして作成します。

```
% ./diff_gcube AB.cube A_B.cube dAB.cube
```

ファイル「dAB.cube」は、相互作用によって引き起こされる（スピン）電子密度の差の cube ファイルであり、ここで「差」は（AB - A_B）を意味します。

50　セルサイズの自動決定

OpenMX では Poission 方程式の解法は FFT を使用しています。そのため、孤立系を計算する場合でも周期境界条件のためにコピーされたイメージが周期的に配置されることになります。孤立系であることを保証するためには最低限、中心セルと隣接セル内の基底関数が重ならないようにスーパーセルを設定することが必要です。この条件の下、スーパーセルを設定すればバンド構造においてバンド分散は生じません。ただし古典的なクーロン相互作用は上記の条件下でスーパーセルを設定したとしても、残っていることに注意して下さい。適切なスーパーセルはユーザー自身で設定する必要がありますが、スーパーセルの設定を自動で行うことも可能です。自動設定では、基底関数のカットオフ半径を考慮して、最小のセルサイズが決定されます。自動設定を行う場合には、キーワード「Atoms.UnitVectors」を入力ファイル中でコメントアウトして下さい。この場合、OpenMX は中心セルの基底関数と隣接セル内の基底関数が重ならないように、適切なセルサイズを自動的に決定し、また要求されるカットオフエネルギーを実現します。決定されたセルベクトルは、次のように標準出力に表示されます。この値を参考にユーザー自身でスーパーセルを設定仕直すことも可能でしょう。

```
<Set_Cluster_UnitCell> automatically determined UnitCell(Ang.)
<Set_Cluster_UnitCell> from atomic positions and Rc of PAOs (margin= 10.00%)
<Set_Cluster_UnitCell> 6.614718 0.000000 0.000000
<Set_Cluster_UnitCell> 0.000000 6.041246 0.000000
<Set_Cluster_UnitCell> 0.000000 0.000000 6.614718

widened unit cell to fit energy cutoff (Ang.)
A = 6.744142 0.000000 0.000000 (48)
B = 0.000000 6.322633 0.000000 (45)
C = 0.000000 0.000000 6.744142 (48)
```

51　開発者のためのインターフェース

開発者のためのインターフェースが用意されています。Kohn-Sham ハミルトニアン、重なり行列、密度行列を利用したい場合には、これらのデータを次のステップで利用可能です。

1. HS.fileout

   入力ファイルにキーワード「HS.fileout」を含めます。

   ```
   HS.fileout                      on      # on|off, default=off
   ```

   計算が正常終了すると、上記のデータがファイル名「*.scfout」に出力されます（「*」は入力ファイルで指定された系の名前「System.Name」）。

2. analysis_example の作成

   ディレクトリ「source」内で、次のコマンドを実行します。

   ```
   % make analysis_example
   ```

   コンパイルが正常終了すると、ディレクトリ「work」内に実行ファイル「analysis_example」が生成されます。

3. ./analysis_example *.scfout

   ディレクトリ「work」に移動して、次のようにしてプログラムを実行します。

   ```
   % ./analysis_example *.scfout
   ```
   もしくは
   ```
   % ./analysis_example *.scfout > HS.out
   ```

   ハミルトニアン、重なり行列、密度行列の要素が「HS.out」に書き出されます。

4. analysis_example の説明

   ファイル「analysis_example」には、これらのデータについての詳しい説明があります。その一部を以下に示します。

   ```
   ***************************************************************
     You can utilize a filename.scfout which is generated by the SCF
     calculation of OpenMX by the following procedure:

      1. Define your main routine as follows:

            int main(int argc, char *argv[])

      2. Include a header file, "read_scfout.h", in your main routine
   ```

(if you want, also in other routines) as follows:

```
#include "read_scfout.h"
```

3. Call a function, read_scfout(), in the main routine as follows:

```
read_scfout(argv);
```

****

52　自動フォース・テスター

OpenMX には多くの機能が実装されていますので、それらが正しく実装されているのか保証することが必要です。実装の信頼性を確認する効果的な方法は、解析的に求めた力と数値計算によって求めた力を比較することです。プログラムに何らかのバグがあれば、それらは互いに異なるでしょう。これを実行するために、次のようにして自動テスターが利用可能です。

逐次計算

```
% ./openmx -forcetest 0
```

並列計算

```
% ./openmx -forcetest 0 "mpirun -np 4 openmx"
```

ここで「0」は、力の整合性を調べる際のエネルギー項のフラグです。フラグの種類は以下に示す 8 種類があり、各数字は次の項目に対応します。

flag	0	1	2	3	4	5	6	7	8
Kinetic	1	0	1	0	0	0	0	0	0
Non-local	1	0	0	1	0	0	0	0	0
Neutral atom	1	0	0	0	1	0	0	0	0
diff Hartree	1	0	0	0	0	1	0	0	0
Ex-Corr	1	0	0	0	0	0	1	0	0
E. Field	1	0	0	0	0	0	0	1	0
Hubbard U	1	0	0	0	0	0	0	0	1

ここで、「1」は力の整合性のチェックに含めるという意味です。ディレクトリ「work/force_example」内には、力の整合性のチェックに使われる 36 個の入力ファイルが保存されています。テスト計算が終了すると、ディレクトリ「work」内にファイル「forcetest.result」が生成されます。比較結果の一部を以下に転載します。

```
force_example/C2_GGA.dat
  flag= 0
  Numerical force= -(Utot(s+ds)-Utot(s-ds))/(2*ds)
  ds=   0.0003000000
  Forces (Hartree/Bohr) on atom 1
                               x              y              z
  Analytic force       -1.676203071292 -1.397113794193 -1.117456296887
  Numerical force      -1.676101156844 -1.397036485449 -1.117288361652
  diff                 -0.000101914447 -0.000077308744 -0.000167935235
```

```
force_example/C2_LDA.dat
  flag= 0
  Numerical force= -(Utot(s+ds)-Utot(s-ds))/(2*ds)
  ......
  ....
```

53　自動メモリーリーク・テスター

OpenMX では、使用メモリは必要に応じて動的に割り当てられています。動的メモリー割り当て（dynamic memory allocation）では、メモリリークが発生する場合があり、MD ステップの増大とともに使用メモリが浪費され、最終的には異常終了する可能性が生じます。

OpenMX を次のように実行することで、メモリリークをチェックすることが可能です。

逐次計算

```
% ./openmx -mltest
```

並列計算

```
% ./openmx -mltest "mpirun -np 4 openmx"
```

ディレクトリ「work/ml_example」に保存されている 13 個のテスト用入力ファイルを用いて、メモリリークが監視されます。この際に、プログラムコード「openmx」中の同じモニターポイントで実際に使用された VSZ および RSS を監視します。計算終了後、ファイル「mltest.result」が生成します。監視された VSZ と RSS は MD ステップの関数として、ファイル「mltest.result」中に次のように出力されていますので、メモリリークが生じているのかどうか確認できます。

```
   1     ml_example/Co4.dat

                  CPU (%)     VSZ (kbyte)    RSS (kbyte)
MD_iter=   1      92.900      49756          15736
MD_iter=   2      84.900      73344          57208
MD_iter=   3      81.200      73344          57212
MD_iter=   4      85.900      98672          82548
MD_iter=   5      82.800      98672          82548
MD_iter=   6      83.800      98672          82548
MD_iter=   7      84.200      98672          82548
MD_iter=   8      84.600      98824          82688
MD_iter=   9      85.000      98824          82688
MD_iter=  10      87.300      98824          82688
MD_iter=  11      87.500      98824          82688
MD_iter=  12      87.400      98824          82688
MD_iter=  13      85.700      98824          82688
MD_iter=  14      84.500      98824          82688
MD_iter=  15      86.100      98824          82688
MD_iter=  16      86.300      98824          82688
MD_iter=  17      86.500      98824          82688
```

```
MD_iter=  18      86.400      98824          82688
MD_iter=  19      86.500      98824          82688
MD_iter=  20      87.500      98824          82688

   2     ml_example/Co4+U.dat

                  CPU (%)     VSZ (kbyte)    RSS (kbyte)
MD_iter=   1      92.800      50048          15924
MD_iter=   2      84.700      73628          57476
MD_iter=   3      84.700      73628          57496
MD_iter=   4      85.300      73628          57496
MD_iter=   5      85.100      98828          82684
MD_iter=   6      85.000      98828          82684
   ......
   ....
```

54　メモリ使用量の解析

「*.memory0」、「*.memory1」、….「*.memory#」のファイルを解析することによってメモリの使用状況が分かります。ここで、「*」はキーワード「System.Name」で指定されるファイル名であり、ファイル拡張子の最後の数（#）は、MPI 並列化におけるプロセス ID です。これらのファイルは、キーワード「memory.usage.fileout」を次のように設定することで出力されます。

```
memory.usage.fileout   on   # default=off, on|off
```

例として、「met.memory0」を下記に示します。

```
Memory: SetPara_DFT: Spe_PAO_XV                                0.01 MBytes
Memory: SetPara_DFT: Spe_PAO_RV                                0.01 MBytes
Memory: SetPara_DFT: Spe_Atomic_Den                            0.01 MBytes
Memory: SetPara_DFT: Spe_PAO_RWF                               0.57 MBytes
Memory: SetPara_DFT: Spe_RF_Bessel                             1.03 MBytes
Memory: SetPara_DFT: Spe_VPS_XV                                0.01 MBytes
Memory: SetPara_DFT: Spe_VPS_RV                                0.01 MBytes
Memory: SetPara_DFT: Spe_Vna                                   0.01 MBytes
Memory: SetPara_DFT: Spe_VH_Atom                               0.01 MBytes
Memory: SetPara_DFT: Spe_Atomic_PCC                            0.01 MBytes
Memory: SetPara_DFT: Spe_VNL                                   0.11 MBytes
Memory: SetPara_DFT: Spe_VNLE                                  0.00 MBytes
Memory: SetPara_DFT: Spe_VPS_List                              0.00 MBytes
.....
....
...
Memory: Poisson: array0                                        4.00 MBytes
Memory: Poisson: array1                                        4.00 MBytes
Memory: Poisson: request_send                                  0.00 MBytes
Memory: Poisson: stat_send                                     0.00 MBytes
Memory: Poisson: request_recv                                  0.00 MBytes
Memory: Poisson: stat_recv                                     0.00 MBytes
Memory: Force: Hx                                              0.00 MBytes
Memory: Force: Hy                                              0.00 MBytes
Memory: Force: Hz                                              0.00 MBytes
Memory: Force: CDM0                                            0.00 MBytes
Memory: Data_Grid_Copy_B2C_1: Work_Array_Snd_Grid_B2C          0.72 MBytes
Memory: Data_Grid_Copy_B2C_1: Work_Array_Rcv_Grid_B2C          0.72 MBytes
Memory: total                                                256.99 MBytes
```

上記のファイルは、入力ファイル「Methane.dat」でキーワード「memory.usage.fileout」を設定、シングルプロセスで実行し、得られたものです。大部分の配列のメモリ使用量は、このファイルに記録されていますが、リストは完全ではないことに注意してください。

55　大規模ファイルのバイナリ形式での出力

大規模計算では、大規模テキスト形式ファイル (例えば、cube ファイル) が生成されます。このようなファイルを出力するための IO アクセスは、IO アクセスが高速でない計算機では、長時間を要します。そのような場合には、大規模ファイルをバイナリ形式で出力することをお勧めします。バイナリ形式での出力はキーワード「OutData.bin.flag」を指定することで実行可能です。

```
OutData.bin.flag    on    # default=off, on|off
```

これにより、大規模ファイルは全てバイナリ形式で出力されます。デフォルトは「off」です。

出力されたバイナリファイルは、ディレクトリ「source」内のプログラム「bin2txt.c」を使用して変換されます。このプログラムは次のようにして、コンパイルされます。

```
gcc bin2txt.c -lm -o bin2txt
```

後処理として、以下のように実行すればバイナリ形式からテキスト形式に変換されます。

```
./bin2txt *.bin
```

この機能は IO アクセスが高速でない計算機では有用でしょう。

56 入力ファイルの例

参考のために、本マニュアル中の計算例に関する入力ファイルがディレクトリ「work」内に保存されています。入力ファイルの一覧を以下に示します。

```
Molecules or clusters
    C60.dat             SCF calc. of a C60 molecule
    C60_DC.dat          DC calc. of a C60 molecule
    CG15c_DC.dat        DC calc. of DNA
    Cr2_CNC.dat         Constrained DFT calc. of a Cr2 dimer
    Doped_NT.dat        SCF calc. of doped carbon nanotube
    Fe2.dat             SCF calc. of a Fe2 dimer
    Gly_NH.dat          Nose-Hoover MD of a glycine molecule
    Gly_VS.dat          Velocity scaling MD of a glycine molecule
    H2O.dat             Geometry opt. of a water molecule
    MCCN.dat            DC calc. of a a multiply connected carbon nanotube
    Methane2.dat        Geometry opt. of a distorted methane molecule
    Methane.dat         SCF calc. of a  methane molecule
    Methane_OO.dat      Orbital optimization of a methane molecule
    Mn12.dat            SCF calc. of a single molecular magnet, Mn12
    Mol_MnO_NC.dat      Non-collinear SCF calc. of a MnO molecule
    Nitro_Benzene.dat   SCF calc. of a nitro benzene molecule under E-field
    Pt13.dat            SCF calc. of a Pt13 cluster
    Pt63.dat            SCF calc. of a Pt63 cluster
    SialicAcid.dat      SCF calc. of a sialic acid molecule
    Valorphin_DC.dat    DC calc. of valorphin molecule
    C2H4_NEB.dat        NEB calc. of C2H4 dimer
    C60_LO.dat          Low-order scaling calc. of a C60 molecule

Bulk
    Cdia.dat            SCF calc. of bulk diamond
    MnO_NC.dat          Non-collinear SCF calc. of bulk MnO
    FeO_NC.dat          Non-collinear SCF calc. of bulk FeO
    CoO_NC.dat          Non-collinear SCF calc. of bulk CoO
    NiO_NC.dat          Non-collinear SCF calc. of bulk NiO
    Crys-NiO.dat        SCF calc. of bulk NiO
    DIA64_Band.dat      SCF calc. of bulk diamond including 64 atoms
    DIA8_DC.dat         DC calc. of bulk diamond including 8 atoms
    DIA64_DC.dat        DC calc. of bulk diamond including 64 atoms
    DIA216_DC.dat       DC calc. of bulk diamond including 216 atoms
```

DIA512_DC.dat	DC calc. of bulk diamond including 512 atoms
DIA512-1.dat	Krylov O(N) calc. of bulk diamond including 512 atoms
Febcc2.dat	SCF calc. of bcc Fe
GaAs.dat	Non-collinear calc. of bulk gallium arsenide
NaCl.dat	SCF calc. of bulk NaCl
NaCl_FC.dat	SCF calc. of bulk NaCl with a Cl-site vacancy
Si8.dat	Geometry opt. of distorted Si bulk
Al-Si111_ESM.dat	ESM calc. of Al-Si interface
Cafcc_FS.dat	Fermi surface calc. of the fcc Ca bulk
Graphite_STM.dat	STM image of graphene
Mnfcc-EvsLC.dat	E vs. lattice constant calc. of the fcc Mn bulk
Si8_NEB.dat	NEB calc. for hydrogen in Si

57　知られている問題点

- 基底関数の過完備性

  fcc、hcp、bcc などの結晶構造をもつ高密度のバルク系において、多くの基底関数を割り当てた場合には、基底関数が過完備になる可能性があります。そのような場合には、数値不安定性のために誤った固有値が生じる可能性が高まります。この過完備性を避けるために、最適化された少数の基底関数を使用することが推奨されます。またこの問題を避ける別の方法として、キーワード「scf.ProExpn.VNA」を「off」に設定することも有効です。

  ```
  scf.ProExpn.VNA      off       # on|off, default = on
  ```

  この場合、実空間の数値グリッドに対するカットオフエネルギをキーワード「scf.energycutoff」によって大き目に設定する必要があるかもしれません。

- SCF の収束が困難なこと

  複雑な（ノンコリニア）磁気的構造、金属的な電子構造、その両者を合わせ持つ場合、またそれらの特徴を持った大規模系では、SCF の収束を得ることがかなり困難です。そのような場合には、電子密度を非常にゆっくりと混合せねばならず、それは残念ながら、収束解を得るために多くの SCF ステップが必要となることを意味します。

- 構造最適化が困難なこと

  分子系のように、相互作用が弱い系では、完全に最適化された構造を得ることは容易でなく、多数の反復ステップが必要となります。構造最適化のデフォルトの収束条件は最大 10^{-4} Hartree/Bohr ですが、そのような場合には、妥協案として収束条件を 10^{-4} から 5×10^{-4} に設定仕直すことも 1 つの方法です。

58　OpenMX フォーラム

OpenMX と ADPACK の技術的問題に関する議論のために、フォーラムが用意されています（http://www.openmx-square.org/forum/patio.cgi）。このフォーラムはOpenMX 利用上のヒントをユーザー間で共有することと、プログラムコード発展のための情報交換の場となることを期待して開設されています。フォーラムの利用上の注意点に関しては、http://www.openmx-square.org/forum/note.html を参照してください。

59 その他

プログラム

本プログラムパッケージは 1 つの makefile と、C 言語と F90 言語で記述されたルーチンで構成されています。

makefile,

また 21 個のヘッダーファイルが含まれます。

exx_debug.h	exx_file_eri.h	exx.h	exx_interface_openmx.h
exx_rhox.h	exx_step2.h	exx_xc.h	Inputtools.h
mimic_sse.h	read_scfout.h	tran_variables.h	exx_def_openmx.h
exx_file_overlap.h	exx_index.h	exx_log.h	exx_step1.h
exx_vector.h	f77func.h	lapack_prototypes.h	openmx_common.h
tran_prototypes.h			

全部で 265 個のプログラムルーティンから構成されており、その内訳を以下に記載します。

add_gcube.c	get_elpa_row_col_comms.f90	SCF2File.c
Allocate_Arrays.c	Get_OneD_HS_Col.c	Set_Aden_Grid.c
analysis_example.c	Get_Orbitals.c	Set_Allocate_Atom2CPU.c
AngularF.c	GR_Pulay_DM.c	Set_Density_Grid.c
Band_DFT_Col.c	Hamiltonian_Band.c	Set_Hamiltonian.c
Band_DFT_Dosout.c	Hamiltonian_Band_NC.c	Set_Initial_DM.c
Band_DFT_kpath.c	Hamiltonian_Cluster.c	Set_Nonlocal.c
Band_DFT_MO.c	Hamiltonian_Cluster_NC.c	Set_OLP_Kin.c
Band_DFT_NonCol.c	Hamiltonian_Cluster_SO.c	Set_Orbitals_Grid.c
bandgnu13.c	init_alloc_first.c	SetPara_DFT.c
Bench_MatMul.c	init.c	Set_ProExpn_VNA.c
BentNT.c	Initial_CntCoes2.c	Set_Vpot.c
bin2txt.c	Initial_CntCoes.c	Set_XC_Grid.c
BroadCast_ComplexMatrix.c	Init_List_YOUSO.c	Show_DFT_DATA.c
BroadCast_ReMatrix.c	Input_std.c	Simple_Mixing_DM.c
check_lead.c	Inputtools.c	Smoothing_Func.c
Cluster_DFT.c	io_tester.c	solve_evp_complex.f90
Cluster_DFT_Dosout.c	iterout.c	solve_evp_real.f90
Cluster_DFT_ON2.c	iterout_md.c	Spherical_Bessel.c
Cont_Matrix0.c	jx.c	test_mpi2.c
Cont_Matrix1.c	Kerker_Mixing_Rhok.c	test_mpi3.c
Cont_Matrix2.c	Krylov.c	test_mpi4.c
Cont_Matrix3.c	KumoF.c	test_mpi.c
Cont_Matrix4.c	lapack_dstedc1.c	test_openmp2.c
Contract_Hamiltonian.c	lapack_dstedc2.c	test_openmp3.c
Contract_iHNL.c	lapack_dstedc3.c	test_openmp.c
Cutoff.c	lapack_dstegr1.c	Tetrahedron_Blochl.c
dampingF.c	lapack_dstegr2.c	Timetool.c
deri_dampingF.c	lapack_dstegr3.c	Total_Energy.c
DFT.c	lapack_dsteqr1.c	TRAN_Add_ADensity_Lead.c
DFTDvdW_init.c	lapack_dstevx1.c	TRAN_Add_Density_Lead.c
diff_gcube.c	lapack_dstevx2.c	TRAN_adjust_Ngrid.c
diff_geo.c	lapack_dstevx3.c	TRAN_Allocate.c
DIIS_Mixing_DM.c	lapack_dstevx4.c	TRAN_Allocate_NC.c
DIIS_Mixing_Rhok.c	lapack_dstevx5.c	TRAN_Apply_Bias2e.c

Divide_Conquer.c
Divide_Conquer_Dosout.c
DosMain.c
Dr_KumoF.c
Dr_RadialF.c
Dr_VH_AtomF.c
Dr_VNAF.c
dtime.c
Eff_Hub_Pot.c
EigenBand_lapack.c
Eigen_lapack2.c
Eigen_lapack.c
Eigen_PHH.c
Eigen_PReHH.c
elpa1.f90
esp.c
EulerAngle_Spin.c
expao.c
exx.c
exx_debug.c
exx_file_eri.c
exx_file_overlap.c
exx_index.c
exx_interface_openmx.c
exx_log.c
exx_rhox.c
exx_step1.c
exx_step2.c
exx_vector.c
exx_xc.c
File_CntCoes.c
Find_CGrids.c
find_Emin0.c
find_Emin2.c
find_Emin.c
find_Emin_withS.c
Force.c
Force_HNL.c
Force_test.c
frac2xyz.c
Free_Arrays.c
FT_NLP.c
FT_PAO.c
FT_ProductPAO.c
FT_ProExpn_VNA.c
FT_VNA.c
Fuzzy_Weight.c
Gaunt.c
Gauss_Legendre.c
Generate_Wannier.c
Generating_MP_Special_Kpt.c
Get_Cnt_dOrbitals.c
Get_Cnt_Orbitals.c
Get_dOrbitals.c
Lapack_LU_inverse.c
LU_inverse.c
Make_Comm_Worlds.c
Make_FracCoord.c
Make_InputFile_with_FinalCoord.c
Maketest.c
malloc_multidimarray.c
MD_pac.c
Memory_Leak_test.c
Merge_LogFile.c
mimic_sse.c
Mio_tester2.c
Mio_tester.c
Mixing_DM.c
mpao.c
mpi_multi_world2.c
mpi_multi_world.c
mpi_non_blocking.c
Mulliken_Charge.c
neb.c
neb_check.c
neb_run.c
Nonlocal_Basis.c
Nonlocal_RadialF.c
Occupation_Number_LDA_U.c
openmx.c
openmx_common.c
Opt_Contraction.c
OpticalConductivityMain.c
Orbital_Moment.c
OutData_Binary.c
OutData.c
Output_CompTime.c
outputfile1.c
Overlap_Band.c
Overlap_Cluster.c
pdb2pao.c
PhiF.c
Poisson.c
Poisson_ESM.c
polB.c
Pot_NeutralAtom.c
PrintMemory.c
PrintMemory_Fix.c
QuickSort.c
RadialF.c
readfile.c
read_scfout.c
ReLU_inverse.c
RestartFileDFT.c
RF_BesselF.c
rmmpi.c
rot.c
Runtest.c
TRAN_Calc_CentGreen.c
TRAN_Calc_CentGreenLesser.c
TRAN_Calc_GridBound.c
TRAN_Calc_Hopping_G.c
TRAN_Calc_OneTransmission.c
TRAN_Calc_SelfEnergy.c
TRAN_Calc_SurfGreen.c
TRAN_Calc_SurfGreen_Sanvito.c
TRAN_Check_Input.c
TRAN_Check_Region.c
TRAN_Check_Region_Lead.c
TRAN_Credit.c
TRAN_Deallocate_Electrode_Grid.c
TRAN_Deallocate_RestartFile.c
TRAN_DFT.c
TRAN_DFT_Dosout.c
TRAN_DFT_NC.c
TRAN_Distribute_Node.c
TRAN_Input_std_Atoms.c
TRAN_Input_std.c
TranMain.c
TranMain_NC.c
TRAN_Output_HKS.c
TRAN_Output_HKS_Write_Grid.c
TRAN_Output_Trans_HS.c
TRAN_Poisson.c
TRAN_Print.c
TRAN_Print_Grid.c
TRAN_Read.c
TRAN_RestartFile.c
TRAN_Set_CentOverlap.c
TRAN_Set_CentOverlap_NC.c
TRAN_Set_Electrode_Grid.c
TRAN_Set_IntegPath.c
TRAN_Set_MP.c
TRAN_Set_SurfOverlap.c
TRAN_Set_SurfOverlap_NC.c
TRAN_Set_Value.c
truncation.c
unit2xyz.c
VH_AtomF.c
VNAF.c
Voronoi_Charge.c
Voronoi_Orbital_Moment.c
XC_CA_LSDA.c
XC_Ceperly_Alder.c
XC_EX.c
XC_PBE.c
XC_PW92C.c
xyz2spherical.c
zero_cfrac.c
zero_fermi.c

さらに、次のライブラリーパッケージがリンクされています。

```
lapack,
blas,
fftw,
MPICH or LAM
omp
```

プログラムパッケージの版権

このプログラムパッケージの配布は、GNU General Public License[59] の方式に従っています。またオリジナルプログラムの作者である尾崎泰助がこのプログラムパッケージのオリジナルバージョンの版権を所有しています。我々はこのプログラムパッケージのユーザーの利用に関して、いかなる保証もできません。しかしプログラムのバグを報告して頂ければ、問題解決に向けて出来る限り利用者と共に協力し作業を行うつもりです。

謝辞

我々の一人（T.O.）は、有益な示唆とコメントをいただいた JRCAT と RICS-AIST の多数の仲間に感謝いたします。我々の一人（T.O.）は、次の国家プロジェクトによって支援を受けました。

- NEDO のナノ機能合成技術プロジェクト (SYNAF-NEDO) [93]
- 科学技術振興機構の計算科学技術活用型特定研究開発推進事業 (ACT-JST) [94]
- 文部科学省の最先端・高性能汎用スーパーコンピュータの開発利用プロジェクト (NAREGI) [95]
- 科学技術振興機構の戦略的創造研究推進事業 (CREST-JST) [96]
- 文部科学省科学研究費補助金「新学術領域研究」コンピューティクスによる物質デザイン [97]
- 文部科学省「HPCI 戦略プログラム (SPIRE)」分野 2 ＜新物質エネルギー創成＞ 計算物質科学イニシアティブ (CMSI) [98]

ご協力に深く感謝申し上げます。

参考文献

[1] P. Hohenberg and W. Kohn, Phys. Rev. **136**, B864 (1964); W. Kohn and L. J. Sham, Phys. Rev. **140**, A1133 (1965).

[2] D. M. Ceperley and B. J. Alder, Phys. Rev. Lett., 45, 566(1980); J. P. Perdew and A. Zunger, Phys. Rev. B **23,** 5048 (1981).

[3] J. P. Perdew and A. Zunger, Phys. Rev. B **23,** 5048 (1981).

[4] J. P. Perdew and Y. Wang, Phys.Rev.B **45**, 13244 (1992).

[5] J. P. Perdew, K. Burke, and M. Ernzerhof, Phys. Rev. Lett. **77**, 3865 (1996).

[6] U. Von. Barth and L. Hedin, J. Phys. C: Solid State Phys. **5**, 1629 (1972).

[7] J. Kübler, K-H. Höck, J. Sticht, and A. R. Williams, J. Phys. F: Met. Phys. **18**, 469 (1988).

[8] J. Sticht, K-H. Höck, and J. Kübler, J. Phys.: Condens. Matter **1**, 8155 (1989).

[9] T. Oda, A. Pasquarello, and R.Car, Phys. Rev. Lett. **80**, 3622 (1998).

[10] A. H. MacDonald and S. H. Vosko, J. Phys. C: Solid State Phys. 12, 2977 (1979).

[11] Ph. Kurz, F. Forster, L. Nordstrom, G, Bihlmayer, and S. Blugel, Phys. Rev. B **69**, 024415 (2004).

[12] R. D. King-Smith and D. Vanderbilt, Phys. Rev. B **47**, 1651 (1993).

[13] G. Theurich and N. A. Hill, Phys. Rev. B **64**, 073106 (2001).

[14] A. I. Liechtenstein, M. I. Katsnelson, V. P. Antropov, and V. A. Gubanov, J. Mag. Mag. Mat. **67**, 65 (1987).

[15] M. J. Han, T. Ozaki, and J. Yu, Phys. Rev. B **70**, 184421 (2004).

[16] M. J. Han, T. Ozaki, and J. Yu, Phys. Rev. B **74**, 045110 (2006).

[17] L. V. Woodcock, Chem. Phys. Lett. **10** ,257 (1971).

[18] S. Nose, J. Chem. Phys. **81**, 511 (1984); S. Nose, Mol. Phys. **52**, 255 (1984); G. H. Hoover, Phys. Rev. A **31**, 1695 (1985)).

[19] G. B. Bachelet, D. R. Hamann, and M. Schluter, Phys. Rev. B **26**, 4199 (1982).

[20] N. Troullier and J. L. Martine, Phys. Rev. B **43**, 1993 (1991).

[21] L. Kleinman and D. M. Bylander, Phys. Rev. Lett. **48,** 1425 (1982).

[22] P. E. Blochl, Phys. Rev. B **41**, 5414 (1990).

[23] I. Morrison, D.M. Bylander, L. Kleinman, Phys. Rev. B **47**, 6728 (1993).

[24] D. Vanderbilt, Phys. Rev. B **41**, 7892 (1990).

[25] H.J. Monkhorst and J.D. Pack, Phys. Rev. B **13**, 5188 (1976).

[26] T. Auckenthaler, V. Blum, H.-J. Bungartz, T. Huckle, R. Johanni, L. Kraemer, B. Lang, and H. Lederer, P. R. Willems, Parallel Computing 27, 783 (2011).

[27] K. Lejaeghere, V. Van Speybroeck, G. Van Oost, and S. Cottenier, arXiv:1204.2733v3. (http://arxiv.org/abs/1204.2733v3)

[28] T. Ozaki, Phys. Rev. B. **67**, 155108, (2003); T. Ozaki and H. Kino, Phys. Rev. B **69**, 195113 (2004).

[29] T. Ozaki and H. Kino, Phys. Rev. B **72**, 045121 (2005).

[30] T. Ozaki, Phys. Rev. B **74**, 245101 (2006).

[31] T.V.T. Duy and T. Ozaki, arXiv:1209.4506v1.

[32] T.V.T. Duy and T. Ozaki, arXiv:1302.6189v1.

[33] S.F. Boys and F. Bernardi, Mol. Phys. **19**, 553 (1970).

[34] S. Simon, M. Duran, and J.J. Dannenberg, J. Chem. Phys. **105**, 11024 (1996).

[35] M. C. Payne, M. P. Teter, D. C. Allan, T. A. Arias and J. Joannopoulos, Rev. Mod. Phys. **64**, 1045 (1992) and references therein.

[36] O. F. Sankey and D. J. Niklewski, Phys. Rev. B. **40**, 3979 (1989)

[37] W. Yang, Phys.Rev.Lett. **66**, 1438 (1991)

[38] P. Ordejon, E. Artacho, and J. M. Soler, Phys. Rev. B. **53**, 10441 (1996)

[39] D. R. Bowler and M. J. Gillan, Chem. Phys. Lett. **325**, 475 (2000).

[40] G. Kresse and J. Furthmeuller, Phys. Rev. B. **54**, 11169 (1996)

[41] G. P. Kerker, Phys. Rev. B **23**, 3082 (1981).

[42] T. A. Arias, M. C. Payne, and J. D. Joannopoulos, Phys. Rev. B **45**, 1538 (1992).

[43] D. Alfe, Comp. Phys. Commun. **118**, 32 (1999).

[44] P. Csaszar and P. Pulay, J. Mol. Struct. (Theochem) **114**, 31 (1984).

[45] J. Baker, J. Comput. Chem. **7**, 385 (1986)

[46] A. Banerjee, N. Adams, J. Simons, R. Shepard, J. Phys. Chem. **89**, 52 (1985)

[47] C. G. Broyden, J. Inst. Math. Appl. 6, 76 (1970); R. Fletcher, Comput. J. 13, 317 (1970); D. Goldrarb, Math. Comp. 24, 23 (1970); D. F. Shanno, Math. Comp. 24, 647 (1970).

[48] P. E. Blochl, O. Jepsen and O. K. Andersen, Phys. Rev. B **49**, 16223 (1994).

[49] A. D. Becke and R. M. Dickson, J. Chem. Phys. **89**, 2993 (1988).

[50] A. Svane and O. Gunnarsson, Phys. Rev. Lett. **65**, 1148 (1990).

[51] Kino's note.

[52] J. Tersoff and D. R. Hamann, Phys. Rev. B **31**, 805 (1985).

[53] G. Henkelman and H. Jonsson, J. Chem. Phys. **113**, 9978 (2000).

[54] T. Ozaki, K. Nishio, and H. Kino, Phys. Rev. **81**, 035116 (2010).

[55] T. Ozaki, Phys. Rev. B **75**, 035123 (2007).

[56] M. Brandbyge, J.-L. Mozos, P. Ordejon, J. Taylor, and K. Stokbro, Phys. Rev. B **65**, 165401 (2002)

[57] G. C. Liang, A. W. Ghosh, M. Paulsson, and S. Datta, Phys. Rev. B. **69**, 115302 (2004).

[58] H. Weng, T. Ozaki, and K. Terakura, Phys. Rev. B **79**, 235118 (2009).

[59] http://www.gnu.org/

[60] http://www.cscs.ch/molekel/

[61] http://www.xcrysden.org/

[62] T. Lis, Acta Crystallogra. B **36**, 2042 (1980).

[63] T. P. Davis T. J. Gillespie, F. Porreca, Peptides **10**, 747 (1989).

[64] A. Goldstein, S. Tachibana, L. I. Lowney, M. Hunkapiller, and L. Hood, Proc. Natl. Acad. Sci. U. S. A. **76**, 6666 (1979).

[65] U. C. Singh and P. A. Kollman, J. Comp. Chem. **5**, 129(1984).

[66] L. E. Chirlian and M. M. Francl, J. Com. Chem. **8**, 894(1987).

[67] B. H. Besler, K. M. Merz Jr. and P. A. Kollman, J. Comp. Chem. **11**, 431(1990).

[68] http://www.webelements.com/

[69] M. Cardona, N. E. Christensen, and G. Gasol, Phys. Rev. B **38**, 1806 (1988).

[70] G. Theurich and N. A. Hill, Phys. Rev. B **64**, 073106 (2001).

[71] *Physics of Group IV Elements and III-V Compounds*, edited by O.Madelung, M.Schulz, and H. Weiss, Landolt-Büornstein, New Series, Group 3, Vol. 17, Pt.a (Springer, Berlin, 1982).

[72] T. Ono and K. Hirose, Phys. Rev. B **72**, 085105 (2005).

[73] W. N. Mei, L. L. Boyer, M. J. Mehl, M. M. Ossowski, and H. T. Stokes, Phys. Rev. B **61**, 11425 (2000).

[74] I. V. Solovyev. A. I. Liechtenstein, K. Terakura, Phys. Rev. Lett. **80**, 5758.

[75] K. Knopfle, L. M. Sandratskii, and J. Kubler, J. Phys:Condens. Matter **9**, 7095 (1997).

[76] I. S. Dhillon and B. N. Parlett, SIAM J. Matrix Anal. Appl. **25**, 858 (2004).

[77] J. J. M. Cuppen, Numer. Math. **36**, 177 (1981); M. Gu and S. C. Eisenstat, SIAM J. Mat. Anal. Appl. **16**, 172 (1995).

[78] N. Mazari and D. Vanderbilt, Phys. Rev. B **56**, 12 847 (1997).

[79] I. Souza, N. Marzari and D. Vanderbilt, Phys. Rev. B **65**, 035109 (2001).

[80] T. Ozaki, Phys. Rev. B **82**, 075131 (2010).

[81] M. Otani and O. Sugino, Phys. Rev. B **73**, 115407 (2006).

[82] O. Sugino, I. Hamada, M. Otani, Y. Morikawa, T. Ikeshoji, and Y. Okamoto, Surf. Sci. **601**, 5237 (2007).

[83] M. Otani, I. Hamada, O. Sugino, Y. Morioka, Y. Okamoto, and T. Ikeshoji, J. Phys. Soc. Jpn. **77**, 024802 (2008).

[84] T. Ohwaki, M. Otani, T. Ikeshoji, and T. Ozaki, J. Chem. Phys. 136, 134101 (2012).

[85] G. Henkelman and H. Jonsson, J. Chem. Phys. **113**, 9978 (2000).

[86] S. Grimme, J. Comput. Chem. **27**, 1787 (2006).

[87] http://www.wannier.org/

[88] http://www.fhi-berlin.mpg.de/th/fhi98md/Murn/readme_murn.html

[89] http://www.openmx-square.org/

[90] http://www.netlib.org/lapack/

[91] http://www.nongnu.org/xmakemol/

[92] http://www.nanotec.es/

[93] http://www.nanoworld.jp/synaf/

[94] http://act.jst.go.jp/

[95] http://ccinfo.ims.ac.jp/nanogrid/

[96] http://www.jst.go.jp/

[97] http://computics-material.jp/index-e.html

[98] http://www.cms-initiative.jp/

索 引