ESopt使用の手引き

ver 1.2.1

平成２３年３月

大阪大学 大学院基礎工学研究科
物質創成専攻 未来物質領域 新物質創製講座

ESopt は東京大学物性研究所において開発が進められた平面波展開法による第一原理電子状態計算プログラム (opt と呼ぶ) を元にして内容を整理したプログラム・パッケージである。物性研からは ISSP FPSOPT として公開版が提供されているが、ESopt は機能を限定した上で Fortran 95 による修正と対角化ルーチンの追加を行っていることから、その派生版であると言える。

旧 opt の特徴として、以下の 2 点が挙げられる。

- ソースコードの可読性の高さ
- 波動関数のもつ全自由度を同時に動かす CG 法の実現

前者の点は ESopt においても引き継がれ、Fortran 77 によるメモリ・アロケーションの困難を回避するために生じた可読性の低下の問題を避ける方向で修正がなされている。また、後者は開発者の荻津格博士 (現米国ローレンス・リバモア研究所) によるものである。比較的多くの CG 法で採用されているバンドごとに最適化を行う方法と異なり、opt における CG 法では波動関数に関する全自由度の同時最適化が実現されている。この CG 法がもつ本来の形式通りのインプリメンテーションが実現されているのは、稀な例である。

この手引きの目的は、ESopt の基礎としている方程式系と実際のプログラムでの実現例を比較することで、第一原理電子状態計算のプログラムを容易に理解し、プログラム動作を理解しながら計算を実行する機会を、平面波基底展開法による計算の初心者に対して提供することである。従って、基礎としている理論の完全な記述は目指さず、必要最低限のものに限りながら、一方で幾つかの具体例において、プログラム上での実現例を紹介することを目指している。

ver. 1.0.0 原子位置最適化ができる Sopt 機能を含む f90 化第一版。

ver. 1.1.0 内部応力計算と結晶構造最適化ができる Allopt 機能を含む第一版の変更版。

ver. 1.2.0 バンド計算機能を含めた変更版。

1 全エネルギーの表式

この節では、平面波展開法における全エネルギーの表式を与える。但し、実際の第一原理計算における全エネルギーの具体的な計算方法を理解することを目的とするため、擬ポテンシャルとして、ノルム保存型の擬ポテンシャルを適用した場合について記述する。$\mathbf{R}_I$, Z_I, $\hat{V}_I^{pseudo}(\mathbf{r}-\mathbf{R}_I)$ を I 番目の原子のもつ原子座標、イオン価数、擬ポテンシャルを表すものとする。

まず、体積 Ω をもつ結晶の全エネルギーの表式として以下のものを用いる。

$$E_{total} = E_{kin} + E_{el-el} + E_{xc} + E_{el-ion} + E_{ion-ion}\,. \tag{1}$$

ここで、E_{kin} は電子の運動エネルギー、E_{el-el} は電子間相互作用のうち電子密度分布間の古典的な Coulomb 相互作用の部分（以下ハートレー・エネルギーと呼ぶ）、E_{xc} は交換相関エネルギー、E_{el-ion} は電子とイオン殻の Coulomb 相互作用、$E_{ion-ion}$ はイオン殻同士の Coulomb 反撥エネルギーである。ただし、イオン殻は球対称ポテンシャルを与えるとした。

それぞれの項は、実空間において以下のように表される。

$$E_{kin} = \sum_{\mathbf{k},n,\sigma} \int_\Omega d\mathbf{r}\ \phi^*_{\mathbf{k},n,\sigma}(\mathbf{r}) \left(-\frac{1}{2}\nabla^2\right) \phi_{\mathbf{k},n,\sigma}(\mathbf{r})\,, \tag{2}$$

$$E_{el-el} = \frac{1}{2}\iint_\Omega d\mathbf{r}\, d\mathbf{r}'\, \frac{n(\mathbf{r})n(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|}\,, \tag{3}$$

$$E_{xc} = \int_\Omega d\mathbf{r}\, \varepsilon_{xc}(n(\mathbf{r}))n(\mathbf{r})\,, \tag{4}$$

$$E_{el-ion} = \sum_{\mathbf{k},n,\sigma}\sum_I \int_\Omega d\mathbf{r}\, \phi^*_{\mathbf{k},n,\sigma}(\mathbf{r})\hat{V}_I^{pseudo}(\mathbf{r}-\mathbf{R}_I)\phi_{\mathbf{k},n,\sigma}(\mathbf{r})\,, \tag{5}$$

$$E_{ion-ion} = \frac{1}{2}\sum_{I,J}\frac{Z_I Z_J}{|\mathbf{R}_I-\mathbf{R}_J|}\,. \tag{6}$$

ここで、一電子密度 $n(\mathbf{r})$ は、

$$n(\mathbf{r}) = \sum_{\sigma=\uparrow,\downarrow} n_\sigma(\mathbf{r})\,, \tag{7}$$

$$n_\sigma(\mathbf{r}) = \sum_{\varepsilon_{\mathbf{k},n,\sigma}\le E_F} |\phi_{\mathbf{k},n,\sigma}(\mathbf{r})|^2\,, \tag{8}$$

と与えられる。$\phi_{\mathbf{k},n,\sigma}(\mathbf{r})$ はサンプル k 点である $\mathbf{k}$ において、スピン σ をもつ n 番目のバンドにおける Kohn-Sham 軌道の波動関数である。そのエネルギーは $\varepsilon_{\mathbf{k},n,\sigma}$ であるとした。E_F はこの仮想的一電子系におけるフェルミ・エネルギーである。

2 Dual space formalism

本パッケージでは、周期的境界条件のもとで、波動関数を平面波基底によって展開して与える。Bloch の定理と Fourier 変換を用いることによって、それぞれの波動関数は平面波の線型和によって以下のように表される。

$$\phi_{\mathbf{k},n,\sigma}(\mathbf{r}) = \sum_{\mathbf{G}} \phi_{\mathbf{k},n,\sigma}(\mathbf{G}) \exp\left(i(\mathbf{k}+\mathbf{G})\cdot\mathbf{r}\right)\,. \tag{9}$$

ここで、$\mathbf{G}$ の和は逆格子の上でとる。

実際の計算では、平面波基底展開特有の周期性を利用した高速化を図るため、分子や原子を計算しようとする場合でも、それを周期的に配置した構造を考えて、全体としての周期性を常に仮定した方法を用いる。この方法に特有の制限に関しても、折に触れて説明することにする。

説明を簡単にするため、周期的構造の基本をなす単位を、単位胞と呼び、N を大きいが有限のある整数として単位胞が N^3 個周期的に配列している構造を考える。つ

まり、単位胞の単位で周期的構造であると同時に、全体としてそれが周期的境界条件のもとにある、という意味で 2 重に周期性が現れる。単位胞の体積を表す Ω_{cell} を導入する。即ち、$\Omega = N^3\Omega_{cell}$ であると考える。

ここで、Fourier 変換について纏めておこう。後に、カットオフを導入することで、$\mathbf{G}$ の有限和が現れるが、まず、$\mathbf{r}$ が連続変数である範囲で変換式を定めておくと便利である。そこで、(9) 式で $\mathbf{G}$ の和は無限和であると考えている。電子密度は結晶の周期と同じ周期をもつと考えてよいことから、密度に関する Fourier 変換は、

$$n_\sigma(\mathbf{G}) = \frac{1}{\Omega_{cell}} \int_{\Omega_{cell}} d\mathbf{r} n_\sigma(\mathbf{r}) \exp\left(-i\mathbf{G}\cdot\mathbf{r}\right) , \tag{10}$$

$$n_\sigma(\mathbf{r}) = \sum_{\mathbf{G}} n_\sigma(\mathbf{G}) \exp\left(i\mathbf{G}\cdot\mathbf{r}\right) , \tag{11}$$

で与えられる。

実際の計算では、実空間と逆格子空間（運動量空間）の両方を用いて計算が行われる。これは、交換相関エネルギーのように実空間で計算するのに適している積分と、運動エネルギーなどのように逆格子空間で計算する方が容易な積分の双方を得る必要があるからである。この方法は、“dual space formalism” と呼ばれている。この方法と高速 Fourier 変換 (FFT) を用いることによって、計算量を大きく減らすことができる。

Kohn-Sham 方程式の解は、逆格子空間において表現された波動関数 $\phi_{\mathbf{k},n,\sigma}(\mathbf{G})$ に関して与えられるエネルギー汎関数の最適化問題として解かれる。そこで、ある波動関数を与えたときに一電子密度を通して得られる各エネルギーを項ごとに与えるために、一電子密度の実空間および波数空間での表現を求めておく必要がある。その具体的な計算の手続きは以下の通りである。

数値的に表現するため、$\mathbf{G}$ は逆格子空間において有限な領域の内部でのみ考える必要がある。より具体的には、展開平面波にあるエネルギーカットオフ E_{cut} を設ける。$|\mathbf{G}|^2 > E_{cut}$ では波動関数の Fourier 成分である $\phi_{\mathbf{k},n,\sigma}(\mathbf{G})$ の値が 0 であるとするのである。このエネルギーカットオフの設定によって、必要とされる逆格子空間の部分空間が定まる。つまり $|\mathbf{G}|^2 \leq E_{cut}$ となる格子点全てがぎりぎり含まれる有限な大きさの逆格子メッシュを定める。（図 1）すると対応して、離散 Fourier 変換を考えた際に実空間上に格子構造が与えられる。この格子を実空間メッシュと呼ぶ。メッシュ上の点を以下では $\mathbf{r}_j$ とする。すると、以下のような計算手順が与えられる。

1. 逆格子空間における波動関数 $\phi_{\mathbf{k},n,\sigma}(\mathbf{G})$ を FFT によって Fourier 逆変換し、(9) 式により与えられる実空間メッシュ上の波動関数 $\phi_{\mathbf{k},n,\sigma}(\mathbf{r}_j)$ を得る。

2. $\phi_{\mathbf{k},n,\sigma}(\mathbf{r}_j)$ から、実空間メッシュ上における電子密度 $n_\sigma(\mathbf{r}_i)$ を (7) 式によって計算する。

3. 実空間メッシュ上の電子密度 $n_\sigma(\mathbf{r}_i)$ から、FFT による Fourier 変換によって電子密度の Fourier 級数を得る。

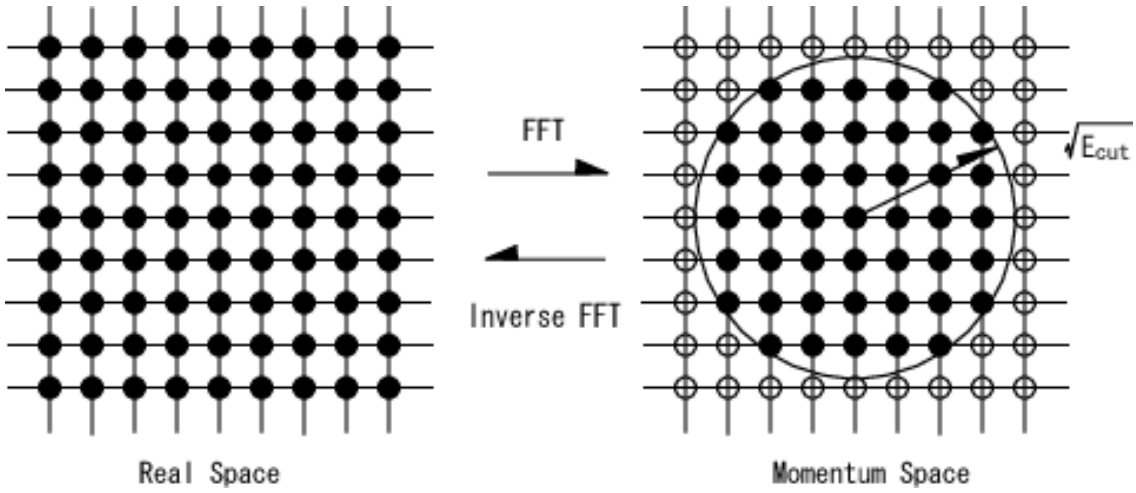

図 1: FFT による Fourier 変換で移り変わる、実空間メッシュと逆格子空間メッシュの対応関係の概念図。逆格子空間（運動量空間）では、$G \leq \sqrt{E_{cut}}$ の範囲にある G 点のみで波動関数が展開される。

最後のステップでは、次式を用いている。

$$n_\sigma(\mathbf{G}) = \sum_j n_\sigma(\mathbf{r}_j) \exp\left(-i\mathbf{G}\cdot\mathbf{r}_j\right) . \tag{12}$$

3　各エネルギーの表式

前節の方法で得られた $\{\phi_{\mathbf{k},n,\sigma}(\mathbf{G})\}$, $\{\phi_{\mathbf{k},n,\sigma}(\mathbf{r}_j)\}$, $\{n_\sigma(\mathbf{r}_j)\}$, $\{n_\sigma(\mathbf{G})\}$ を用いて以下のように具体的な計算が行われる。以下のエネルギーの表式は、すべて単位胞当たりのエネルギーである。

3.1　運動エネルギー

この項は、逆格子空間で計算される。単位胞当たりの運動エネルギーを $\bar{E}_{kin}$ とすることで、

$$\begin{aligned}
\bar{E}_{kin} &= \sum_{\mathbf{k},n,\sigma}\sum_{\mathbf{G},\mathbf{G}'}\int_{\Omega_{cell}} d\mathbf{r}\ \frac{|\mathbf{k}+\mathbf{G}|^2}{2}\exp\left(i(\mathbf{G}-\mathbf{G}')\cdot\mathbf{r}\right)\phi^*_{\mathbf{k},n,\sigma}(\mathbf{G}')\phi_{\mathbf{k},n,\sigma}(\mathbf{G}) \\
&= \frac{\Omega_{cell}}{2}\sum_{\mathbf{k},n,\sigma}\sum_{\mathbf{G}}|\mathbf{k}+\mathbf{G}|^2|\phi_{\mathbf{k},n,\sigma}(\mathbf{G})|^2 .
\end{aligned} \tag{13}$$

エネルギーカットオフを導入した段階で、$\mathbf{G}$ の和を有限和に書き直すことで、この表式を用いてよい。この計算は、実空間で差分近似などによって評価しても計算量が大きく増加することはないが、指数関数に対しては微分演算の結果が数の積で表現できることから、波数空間で行う方が取り扱いが簡単である。

3.2 ハートレー・エネルギー

この項も、逆格子空間で計算される。同じく、単位胞当たりのハートレー・エネルギーを $\bar{E}_{el-el}$ とする。

$$\begin{aligned}
\bar{E}_{el-el} &= \frac{1}{N^3} E_{el-el} \\
&= \frac{1}{2}\int_{\Omega_{cell}} d\mathbf{r} \int_{\Omega} d\mathbf{r}' \, \frac{n(\mathbf{r})n(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} \\
&= \frac{1}{2}\int_{\Omega_{cell}} d\mathbf{r} \int_{\Omega} d\mathbf{s} \frac{1}{|\mathbf{s}|} \sum_{\mathbf{G},\mathbf{G}',\sigma,\sigma'} n_\sigma(\mathbf{G}) n_\sigma(\mathbf{G}') \exp\left(i\mathbf{G}\cdot\mathbf{r} + i\mathbf{G}'\cdot(\mathbf{r}-\mathbf{s})\right) \\
&= \frac{\Omega_{cell}}{2} \sum_{\mathbf{G},\sigma,\sigma'} n_\sigma(\mathbf{G}) n_\sigma(-\mathbf{G}) \int_{\Omega} d\mathbf{s} \frac{1}{|\mathbf{s}|} \exp\left(i\mathbf{G}\cdot\mathbf{s}\right).
\end{aligned}$$

ここで、$\mathbf{G} \to \mathbf{0}$ で生じる発散は、$E_{ion-ion}$, E_{el-ion} の各項にも現れ、系が中性の時には完全に打ち消し合うことが分かっている。そのため、各項での発散部分を予め除いておく。$\bar{E}_{el-el}$ では、$\mathbf{G} = \mathbf{0}$ の項が対応するため、この項を除くことにする。さらに、$N \to \infty$ と考えて $\mathbf{G} \neq \mathbf{0}$ のときに、

$$\int_{\Omega} d\mathbf{s} \frac{1}{|\mathbf{s}|} \exp\left(i\mathbf{G}\cdot\mathbf{s}\right) = \frac{4\pi}{G^2},$$

であることを用いて、また $n(\mathbf{r})$ が実であることから $n_\sigma(-\mathbf{G}) = n_\sigma^*(\mathbf{G})$ が導かれることを用いると、$n(\mathbf{G}) = \sum_\sigma n_\sigma(\mathbf{G})$ と記載することにして、次式を得る。

$$\bar{E}_{el-el} = \frac{\Omega_{cell}}{2} {\sum_{\mathbf{G}}}' \frac{4\pi |n(\mathbf{G})|^2}{G^2}. \tag{14}$$

ここで、$\sum'_{\mathbf{G}}$ は $\mathbf{G} = \mathbf{0}$ を除くことを示している。

この計算は、実空間で行うと 6 次元空間積分となるため、実空間メッシュの格子点数 N_{mesh} に対して、$O(N_{mesh}^2)$ の演算が必要となる。それに対して、逆格子空間で行った場合には、FFT の計算に必要となる演算量 $O(N_{mesh} \log N_{mesh})$ が全体としての演算量を決める。そこで、この計算は逆格子空間で行った方が容易であると言える。

3.3 交換相関エネルギー

交換相関エネルギー密度 ε_{xc} にはいくつかの表式があるが、本パッケージで提供しているのは、Perdew-Zunger の表式 (PZ81) である。この表式はその後の Perdew-Wang の表式 (PW92) と比較した場合には、両者とも基本的に Ceperley-Alder の量子モンテカルロ計算の結果を解析的に表現することを基本としているが、PW92 では Vosko らのスピン偏極依存性を取り入れ、モンテカルロ計算の結果をパラメータ

化する際に個々の計算結果の誤差を考慮して重み付けを導入している点などの差異がある。

$$\bar{E}_{xc} = \int_{\Omega_{cell}} d\mathbf{r} \varepsilon_{xc}\left(n(\mathbf{r})\right) n(\mathbf{r}) \,. \tag{15}$$

交換相関エネルギー密度 ε_{xc} は、交換エネルギー密度 ε_x と相関エネルギー密度 ε_c に分けることができる。

$$\varepsilon_{xc} = \varepsilon_x + \varepsilon_c \,. \tag{16}$$

スピン依存性は本パッケージの現バージョンでは考慮されていない。交換エネルギー密度としては、

$$\begin{aligned} \varepsilon_x(r_s) &= -\frac{3}{4\pi r_s}\left(\frac{9\pi}{4}\right)^{\frac{1}{3}} \\ r_s &= -\left(\frac{3}{4\pi n}\right)^{\frac{1}{3}} , \end{aligned} \tag{17}$$

が用いられる。相関エネルギー密度の表式は省略する。

3.4 電子ーイオン間相互作用

イオンの荷電は古典的な電荷分布として扱う。それを、擬ポテンシャルとして問題に反映する。本パッケージでは、E_{el-ion} は擬ポテンシャル $\hat{V}_I^{pseudo}$ を用いて計算される。内殻の電子の効果は擬ポテンシャルに含まれるため、擬ポテンシャル法では価電子のみを顕わに扱うことになる。或いは、内殻電子の電荷分布は固定されているとする近似を採用していることになる。本パッケージで採用している擬ポテンシャルによるエネルギーは、局所ポテンシャルと非局所ポテンシャルの和で表される。

$$\begin{aligned} E_{el-ion} &= E_{el-ion}^{local} + E_{el-ion}^{non-local} \\ &= \sum_I \int_{\Omega_c ell} d\mathbf{r}\, n(\mathbf{r}) V_I^{local}\left(|\mathbf{r}-\mathbf{R}_I|\right) \\ &+ \sum_{\mathbf{k},n,\sigma} \sum_{I,l\neq l_0} \int_{\Omega_c ell} d\mathbf{r}\, \phi^*_{\mathbf{k},n,\sigma}(\mathbf{r}) \delta\hat{V}_I^l(\mathbf{r}-\mathbf{R}_I) \phi_{\mathbf{k},n,\sigma}(\mathbf{r}) \,. \end{aligned} \tag{18}$$

ここで、l は軌道角運動量を表す。本パッケージに含まれている擬ポテンシャルでは、$l = 0, 1, 2$ のどれか一つ（それを l_0 とする）の寄与を局所ポテンシャルに含め、その他が非局所ポテンシャルとなることを仮定している。

$$V_I^{local}(r) = V_I^{core} + \delta V_I^{l_0} \,. \tag{19}$$

ここで、V_I^{core} は内殻のポテンシャルを表す。

逆格子空間における E_{el-ion} は次のようになる。

$$E_{el-ion} = \Omega_{cell} \sum_{\mathbf{G}}{}' \sum_I S_I(\mathbf{G}) n(\mathbf{G}) V_I^{local}(\mathbf{G}) + \Omega_{cell} n(\mathbf{0}) \sum_I \alpha_I$$

$$+ \quad \Omega_{cell} \sum_{\mathbf{k},n,\sigma} \sum_{\mathbf{G},\mathbf{G}'} \sum_{I,l\neq l_0} S_I(\mathbf{G}-\mathbf{G}')\phi^*_{\mathbf{k},n,\sigma}(\mathbf{G})\delta\hat{V}^l_I(\mathbf{k}+\mathbf{G},\mathbf{k}+\mathbf{G}')\phi_{\mathbf{k},n,\sigma}(\mathbf{G}') \,. \tag{20}$$

$$S_I(\mathbf{G}) \;=\; \exp\left(-i\mathbf{G}\cdot\mathbf{R}_I\right) . \tag{21}$$

ここで、$V^{local}_I(\mathbf{G})$ は I 番目の原子における擬ポテンシャルの局所部分の Fourier 成分、$S_I(\mathbf{G})$ は構造因子を示す。E_{el-el}, $E_{ion-ion}$, E_{el-ion} の $\mathbf{G}=\mathbf{0}$ での発散は、和をとることによって完全に打ち消し合う。しかしながら、擬ポテンシャル法を用いることによって有限の値が残り、全エネルギーにずれが生じる。この $\mathbf{G}\to\mathbf{0}$ の極限での全エネルギーのずれは、$\mathbf{G}\to\mathbf{0}$ における E^{local}_{el-ion} と I 番目の原子における生の Coulomb ポテンシャル $-4\pi Z_I/\Omega_{cell}G^2$ との差によるものである。これは α_I として次のように表される。

$$\alpha_I = \lim_{\mathbf{G}\to\mathbf{0}} \left(S_I(\mathbf{G})V^{local}_I(\mathbf{G}) - \frac{4\pi Z_I}{\Omega_{cell}G^2} \right) . \tag{22}$$

エネルギーの非局所部分からの寄与を計算するためには、平面波の数 N に対して、$O(N^2)$ の演算量が必要である。しかしながら、Kleinman と Bylander による近似 (K-B 近似) を用いることによって、演算量を $O(N)$ にすることが可能である。ここでは、この K-B 近似を適用した場合の表式を用いる。

$$\delta\hat{V}^{non-local} = \sum_{l,m} \frac{|\delta\hat{V}^l\phi_{l,m}\rangle\langle\phi_{l,m}\delta\hat{V}^l|}{\langle\phi_{l,m}|\delta\hat{V}^l|\phi_{l,m}\rangle} \,. \tag{23}$$

ここで $\phi_{l,m}$ は軌道角運動量 l、磁気量子数 m をもつ擬波動関数である。(23) 式は $\phi_{l,m}$ が完全系を張っていれば正しい式であるが、K-B 近似ではこれを擬イオンに対する固有状態として求められた擬波動関数にとる。一般的な系に対しては、従って近似である。多くの系においては物理量に悪影響を与えないことが知られている。しかし、幾つかの場合にはこの近似による副作用が現れることも知られている。一般的な表式における非局所ポテンシャルは、本来動径座標に関しては局所な表示になっており、その意味で semi-local なポテンシャルと呼ばれている。ところが、K-B 近似を行うと、動径方向に関しても非局所なポテンシャルになってしまう。このため Wronsky の定理が満たされなくなることが知られている。そこで、K-B 近似したポテンシャルにおいては、ノードをもつ偽の束縛状態が、ノードを持たない真の束縛状態より低いエネルギー固有状態として現れることがあり得る。このような異常な束縛状態は ghost states と呼ばれている。

K-B 近似により、非局所ポテンシャル $\delta\hat{V}^l(\mathbf{k}+\mathbf{G},\mathbf{k}+\mathbf{G}')$ は以下のように表される。

$$\begin{aligned}
\delta\hat{V}^l(\mathbf{k}+\mathbf{G},\mathbf{k}+\mathbf{G}') &= \frac{1}{\Omega}\int_\Omega d\mathbf{r}\, \exp\left(-i(\mathbf{k}+\mathbf{G})\cdot\mathbf{r}\right)\delta\hat{V}^l(r)\hat{P}_l \exp\left(i(\mathbf{k}+\mathbf{G})\cdot\mathbf{r}\right) \\
&= \frac{4\pi}{\Omega}(2l+1)P_l\left(\cos\theta_{\mathbf{k}+\mathbf{G},\mathbf{k}+\mathbf{G}'}\right) \\
&\quad \times \int_0^\infty d\mathbf{r}\, r^2 j_l(|\mathbf{k}+\mathbf{G}|r) j_l(|\mathbf{k}+\mathbf{G}'|r)\delta\hat{V}^l(r)
\end{aligned}$$

$$= \frac{4\pi}{\Omega}(2l+1)P_l\left(\cos\theta_{\mathbf{k}+\mathbf{G},\mathbf{k}+\mathbf{G}'}\right)\frac{\chi_I^l(\mathbf{k}+\mathbf{G})\chi_I^l(\mathbf{k}+\mathbf{G}')}{X_I^l}\,, \quad (24)$$

$$\cos\theta_{\mathbf{k}+\mathbf{G},\mathbf{k}+\mathbf{G}'} = \frac{(\mathbf{k}+\mathbf{G})\cdot(\mathbf{k}+\mathbf{G}')}{|\mathbf{k}+\mathbf{G}||\mathbf{k}+\mathbf{G}'|}\,, \quad (25)$$

$$\chi_I^l(\mathbf{k}+\mathbf{G}) = \int_0^\infty d\mathbf{r}\, r^2 j_l(|\mathbf{k}+\mathbf{G}|r)\delta\hat{V}^l(r)R_l(r)\,, \quad (26)$$

$$X_I^l = \int_0^\infty d\mathbf{r}\, r^2\delta\hat{V}^l(r)|R_l(r)|^2\,. \quad (27)$$

ここで、$R_l(r)$ は擬ポテンシャル $\delta\hat{V}^l(r)$ によって表される擬波動関数 $\phi_{l,m} = Y_{l,m}(\theta,\phi)R_l(r)$ の動径部分である。

3.5 イオンーイオン間相互作用

この項は、Ewald 和の方法によって計算される。

$$\begin{aligned} E_{ion-ion} &= \frac{1}{2}\frac{1}{N_{cell}}\sum_{I,J}\frac{Z_I Z_J}{|\mathbf{R}_I-\mathbf{R}_J|} \\ &= \frac{1}{2}\sum_{n,m,\mathbf{L}}\frac{Z_n Z_m}{|\mathbf{d}_n-\mathbf{d}_m-\mathbf{L}|} \\ &= \frac{1}{2}\sum_{n,m,\mathbf{L}} Z_n Z_m\left(\frac{\mathrm{erfc}(\kappa|\mathbf{d}_n-\mathbf{d}_m-\mathbf{L}|)}{|\mathbf{d}_n-\mathbf{d}_m-\mathbf{L}|}+\frac{\mathrm{erf}(\kappa|\mathbf{d}_n-\mathbf{d}_m-\mathbf{L}|)}{|\mathbf{d}_n-\mathbf{d}_m-\mathbf{L}|}\right) \quad (28) \end{aligned}$$

ここで、$\mathbf{L}$ は周期系における並進ベクトルである。原子の座標 $\mathbf{R}_I$ は単位胞内で原子の位置を指定するベクトル $\mathbf{d}_n$ によって置き換えられ、$\mathbf{R}_I = \mathbf{d}_n + \mathbf{L}$ で表される。(28) 式の第一項は実空間で計算され、第二項は逆格子空間で計算される。$f(r) = \mathrm{erf}(\kappa r)$ の Fourier 係数は次のように計算される。

$$\begin{aligned} f(\mathbf{G}) &= \lim_{\mu\to 0}\frac{1}{\sqrt{\Omega_{cell}}}\int d\mathbf{r}\, f(r)\exp\left(-i\mathbf{G}\cdot\mathbf{r}-\mu r\right) \\ &= \frac{4\pi}{\Omega_{cell}}\frac{e^{-(G/2\kappa)^2}}{G^2}\,. \quad (29) \end{aligned}$$

このとき、Fourier 係数 $f(\mathbf{G})$ の $\mathbf{G}\to\mathbf{0}$ での振る舞いは、

$$\begin{aligned} f(\mathbf{G}) &\simeq \frac{4\pi}{\Omega_{cell}}\left[1-\left(\frac{G}{2\kappa}\right)^2+O(G^4)\right]\frac{1}{G^2} \\ &= \frac{4\pi}{\Omega_{cell}}\left[\frac{1}{G^2}-\frac{1}{2\kappa}+O(G^2)\right] \quad (30) \end{aligned}$$

となる。ここで、第一項は E_{el-el}, E_{el-ion} と打ち消しあい、第二項のみが残る。それゆえ、発散項を取り除いた $E_{ion-ion}$ は Ewald パラメータ γ_{ewald} として以下のように与えられる。

$$\begin{aligned} \gamma_{ewald} &= \sum_{n,m} Z_n Z_m\left[\frac{4\pi}{\Omega_{cell}}\left(\sum_{\mathbf{G}}{}'\frac{e^{-(G/2\kappa)^2}}{G^2}e^{i\mathbf{G}\cdot(\mathbf{d}_n-\mathbf{d}_m)}-\frac{1}{4\kappa}\right)\right. \\ &+ \left.\sum_{\mathbf{L}}{}'\frac{\mathrm{erf}(\kappa|\mathbf{d}_n-\mathbf{d}_m-\mathbf{L}|)}{|\mathbf{d}_n-\mathbf{d}_m-\mathbf{L}|}-\frac{2\kappa}{\sqrt{\pi}}\delta_{n,m}\right]\,. \quad (31) \end{aligned}$$

単位胞内に含まれる原子数が多くなると、Ewald 和の計算時間は無視できなくなる。そこで、計算量を節約するために、以下の考えに基づいて収束に必要な和の取り方を最適化する。(31) 式において、実空間と逆格子空間のそれぞれにおける和にはカットオフがあり、およそ、$1/\kappa$, 2κ の程度に見積もることができる。実空間において、単位胞の一辺が L であるとすると、逆格子ベクトルの長さは、$G_0 = 2\pi/L$ である。両空間での γ_{ewald} の収束に必要な格子ベクトルの最大値は、カットオフ半径と $L_{max} = LN_{Rmax} \propto 1/\kappa$ と $G_{max} \propto G_0 N_{Gmax} \propto 2\kappa$ の関係がある。よって、両空間においてとられる格子点の総数は、$N_{total} \propto N_{Rmax}^3 + N_{Gmax}^3 \propto \frac{1}{(\kappa L)^3} + \left(\frac{\kappa L}{\pi}\right)^3$ と見積もられ、$\kappa \simeq \frac{\pi}{L}$ にとれば最小の計算量で計算できることが分かる。単位胞が直方体の場合には、評価式は $\kappa \simeq \frac{\pi}{(L_x L_y L_z)^{1/3}}$ となる。

4 Hellmann-Feynman Force

この節では、原子に働く力の表式を与える。I 番目の原子に働く力 F_I は、全エネルギーを原子座標 $\mathbf{R}_I$ で微分することで得られる。

$$\begin{aligned} \mathbf{F}_I &= -\frac{dE}{d\mathbf{R}_I} \\ &= -\sum_i \left[\langle\phi_i| \left(\frac{d}{d\mathbf{R}_I} H\right) |\phi_i\rangle + \left(\frac{d}{d\mathbf{R}_I}\langle\phi_i|\right) H|\phi_i\rangle + \langle\phi_i|H\left(\frac{d}{d\mathbf{R}_I}|\phi_i\rangle\right)\right] \end{aligned} \quad (32)$$

ここで、$i(= (\mathbf{k}, n, \sigma))$ は、Kohn-Sham 波動関数を指定する量子数の組を表すものとする。H を Kohn-Sham Hamiltonian とすると、全エネルギーは今の場合、$E = \sum_i \langle\phi_i|H|\phi_i\rangle$ で表される。それゆえ、波動関数からの寄与が存在するため、一般には計算量が非常に大きくなる。しかしながら、波動関数 ϕ_i が H の規格化された固有状態の完全系を与えるとすれば、次のように (32) 式の第二項の和は 0 となる。

$$\begin{aligned} &\sum_i \left[\left(\frac{d}{d\mathbf{R}_I}\langle\phi_i|\right) H|\phi_i\rangle + \langle\phi_i|H\left(\frac{d}{d\mathbf{R}_I}|\phi_i\rangle\right)\right] \\ &= \sum_i \varepsilon_i \frac{d\langle\phi_i|\phi_i\rangle}{d\mathbf{R}_I} = 0 \end{aligned} \quad (33)$$

それゆえ、原子に働く力の表式は以下のようになる。

$$\mathbf{F}_I = -\sum_i \langle\phi_i| \frac{dH}{d\mathbf{R}_I} |\phi_i\rangle = -\frac{\partial E}{\partial \mathbf{R}_I} \quad (34)$$

(34) 式は、各波動関数がハミルトニアンの固有状態であるとき、エネルギーの原子座標に関する偏微分が原子に働く物理的な力を与えることを意味している。これは Hellmann-Feynman の定理と呼ばれており、この定理を用いることによって、原子に働く物理的な力を簡単に計算することができる。しかしながら、この定理を適用するためには、全エネルギーが基底状態の値に充分収束していなければならない。また、原子に働く力を精度良く計算することは、全エネルギーを精度良く計算する

ことに比べて困難になる。その理由は、全エネルギーの誤差が波動関数の誤差の 2 次のオーダーであるのに対して、力の誤差は波動関数の誤差の 1 次のオーダーだからである。

波動関数を平面波基底で展開した場合、原子座標が顕わに含まれないという性質がある。そのため、エネルギーを原子座標で微分したとき、ノルム保存型擬ポテンシャルを用いた場合の力の表式としては、顕わに原子座標依存性をもつ E_{el-ion}, $E_{ion-ion}$ の 2 項のみが有限の値をもち、他の項は 0 となる。

$$\mathbf{F}_I = -\frac{\partial}{\partial \mathbf{R}_I}\left(E_{el-ion} + E_{ion-ion}\right). \tag{35}$$

ここで、$\dfrac{\partial E_{el-ion}}{\partial \mathbf{R}_I}$ は局所部分と非局所部分の 2 つの項で記述される。

5 Quantum Stress

内部応力の表式は力の表式と同様に変分法で与えられる。体積が Ω_{cell} である単位胞をもつ多電子系を考える。全エネルギーは $E(\{\psi_{\mathbf{k},i}(\mathbf{r})\},\{\mathbf{R}_I\})$ で与えられる。変分原理は、変量である $\{\psi_{\mathbf{k},i}(\mathbf{r})\},\{\mathbf{R}_I\}$ に関して最適化することで、基底状態エネルギーが $E(\{\psi_{\mathbf{k},i}(\mathbf{r})\},\{\mathbf{R}_I\})$ の最小値として与えられることを示している。内部応力を得るには、2 階テンソルである回転自由度を落とした対称歪みテンソル ε を与える必要がある。無限小の ε の変化に対するスケーリングから、以下を得る。

$$\begin{aligned}
\mathbf{R}_I &\rightarrow \mathbf{R}'_I = (1+\varepsilon)\mathbf{R}_I, \\
\psi_{\mathbf{k},i}(\mathbf{r}) &\rightarrow \psi_{\mathbf{k},i}(\mathbf{r}') = \det(1+\varepsilon)^{\frac{1}{2}}\psi_{\mathbf{k},i}(\mathbf{r}).
\end{aligned}$$

基底状態変化は、

$$E_{tot} \rightarrow E(\varepsilon) = E_{tot} + \Delta E(\varepsilon).$$

このエネルギーシフト $\Delta E(\varepsilon)$ は ε に関して展開される。応力テンソル σ はこの展開の 1 次の項の展開係数として与えられる。

$$\begin{aligned}
\Delta E(\varepsilon) &= -\mathrm{Tr}(\sigma\varepsilon)\Omega_{cell} + O(\varepsilon^2), \\
\sigma_{\alpha\beta} &= -\frac{1}{\Omega_{cell}}\frac{\partial E(\varepsilon)}{\partial \varepsilon_{\alpha\beta}}\bigg|_{\varepsilon\rightarrow 0}.
\end{aligned} \tag{36}$$

もし、歪みと応力が等方的であれば、よく知られた $P = -\dfrac{dE}{d\Omega_{cell}}$ という関係式を得る。

対称テンソルである歪みテンソル $\varepsilon_{\alpha\beta}$ に関して、一次の範囲で $\mathbf{G}$ は $(1-\varepsilon)\mathbf{G}$ となり、Ω_{cell} は $\det(1+\varepsilon)\Omega_{cell}$ となる。$\Omega_{cell}n(\mathbf{G})$ と $S_I(\mathbf{G})$ は不変であるので、以下の表式を得ることになる。

$$\sigma_{\alpha\beta} = \sigma^{kin}_{\alpha\beta} + \sigma^{el-el}_{\alpha\beta} + \sigma^{xc}_{\alpha\beta} + \sigma^{local}_{\alpha\beta} + \sigma^{non-local}_{\alpha\beta} + \sigma^{\alpha}_{\alpha\beta} + \sigma^{ewald}_{\alpha\beta}.$$

$$\sigma_{\alpha\beta}^{kin} = 2\sum_{\mathbf{k},\mathbf{G},i} |\psi_i(\mathbf{k}+\mathbf{G})|^2 (\mathbf{k}+\mathbf{G})_\alpha (\mathbf{k}+\mathbf{G})_\beta,$$

$$\sigma_{\alpha\beta}^{el-el} = -\frac{1}{2}{\sum_{\mathbf{G}}}' \frac{4\pi|n(\mathbf{G})|^2}{|\mathbf{G}|^2}\left(\frac{2\mathbf{G}_\alpha\mathbf{G}_\beta}{|\mathbf{G}|^2} - \delta_{\alpha\beta}\right),$$

$$\sigma_{\alpha\beta}^{xc} = \delta_{\alpha\beta}\frac{1}{\Omega_{cell}}\int_{\Omega_{cell}} d\mathbf{r} n(\mathbf{r})(\mu_{xc}(n) - \varepsilon_{xc}(n)),$$

$$\sigma_{\alpha\beta}^{local} = -{\sum_{\mathbf{G},I}}' n^*(\mathbf{G}) S_I(\mathbf{G}) \frac{\partial V_I^{local}(\mathbf{G})}{\partial \varepsilon_{\alpha\beta}},$$

$$\sigma_{\alpha\beta}^{non-local} = -2\sum_{\mathbf{k},i}\sum_{\mathbf{G},\mathbf{G}'}\sum_{I,l} S_I(\mathbf{G}-\mathbf{G}')\psi_{\mathbf{k},i}^*(\mathbf{G})\psi_{\mathbf{k},i}(\mathbf{G}')\frac{\partial \delta \hat{V}_I^l(\mathbf{k}+\mathbf{G},\mathbf{k}'+\mathbf{G}')}{\partial \varepsilon_{\alpha\beta}}$$

$$\sigma_{\alpha\beta}^{\alpha} = \delta_{\alpha\beta} n^*(\mathbf{0}) \sum_I \alpha_I,$$

$$\sigma_{\alpha\beta}^{ewald} - \frac{1}{\Omega_{cell}}\frac{\partial \gamma_{ewald}}{\partial \varepsilon_{\alpha\beta}}.$$

ここで原子位置に関して参照座標を $\mathbf{q}_I$ として導入したが、$\mathbf{R}_I(t) = (1+\varepsilon)\mathbf{q}_I(t)$ である。

6 ESoptの構造

現バージョンの ESopt は以下のような構成となっている。なお、将来変更される可能性がある。`ESopt.tar.gz` を展開すると、`ESopt` というディレクトリーができる。その下に、`INPUT_DATA`, `mknon`, `commons`, `params` というサブディレクトリができる。以下それぞれのディレクトリについて説明する。

6.1 ESopt

プログラムファイル (`*.f90`), `makefile` と関連した実行ディレクトリ作成スクリプトファイルが含まれている。`ESopt` では入力ファイルを以下に説明する `INPUT_DATA` において作成したあとで、このメインディレクトリで `make` を実行し、実行ディレクトリを作る。そののち、実行ディレクトリ（その名称は `makefile` で指定する）において実際の実行を行うことになる。

6.2 INPUT_DATA

計算に必要な入力ファイル群が格納されている。基本的に、ユーザが設定しなければならないのは、CORD というファイルと、*.CNTL というファイルである。CORD では、コンパイル時に必要なパラメータ群（物質の構造や計算条件）を与え、*.CNTL は実行時に読まれるパラメータ群（動作モードの切り替えなど）を与える。擬ポテンシャルのデータや、k 点サンプリングのやり方を与えるデータ、バンド計算時に計算される波数を指定するファイルもこのディレクトリにある。

6.3 mknon

計算に必要なパラメータセットを計算する二つのサブプログラムが格納されている。このプログラム群を利用することにより、ユーザは本質的に独立な入力変数のみを設定すれば良いようになっている。一つめのサブプログラム、kbexe では、FFT mesh 数や、フーリエコンポーネントの数が、カットオフとユニットセルサイズから自動的に計算される。また、電子の状態数は原子の価数の総和から計算される。これらの、システム依存のパラメータのうち、プログラムの実行時に配列サイズ等の決定のため必要なものは、parameters.data というファイルにフォートランの unformatted 形式で出力される。それ以外のものは、input.dat という同じく unformatted 形式で出力される。また、二つめのサブプログラム mknon では、運動量空間での FFT メッシュ点（直方体型に分布する）と、カットオフによって球形に切り抜かれた範囲内にあるフーリエコンポーネントとの対応関係のリストを求め、gpt_list というファイルに unformatted 形式で出力する。これらのプログラムは make を実行することにより自動的に実行される。

6.4 commons と params

本パッケージにおいては、サブルーチン間での変数と配列の引き渡しを原則として以下のようにしている。配列に関しては、module 中で public 宣言したグローバル配列を定義して各サブルーチンで参照する。一方変数の一部は、common 文による参照を行っている。これは、ベースにした Fortran 77 プログラムからの変更が完全ではないためである。mknon で計算されたパラメータにより、グローバル配列のアロケーションを行う際に配列サイズが決定されるが、mknon/parameters.data を読み込んで参照するための module が parameters.f90 であり、一方配列参照のためのものが、globalarray.f90 である。commons においては、サブルーチン間で共通に参照する変数を幾つかのヘッダーファイルの形で提供している。一方、params においては、LDA ポテンシャルのパラメタのような物質依存でないパラメータが格納されている。

7　プログラム実行の概要

プログラムの実行に関する流れは以下のようになっている。

1. INPUT_DATA/CORD により、計算するシステムに関するパラメータ（原子座標、種類、ユニットセルサイズ）および計算条件（平面波のカットオフ半径、k 点サンプリングの種類、など）を設定する。

2. makefile の先頭にある DESTDIR=*destination* の *destination* の部分が実行ディレクトリ名になるため、ここを変更する。なお、実習ではマシンを指定するため、マシン依存性に関する記述は省略。

3. make する。

4. cd *destination* する。

5. 実行モード切替用入力ファイル、SOPT.CNTL, ALLOPT.CNTL, FILE_IO.CNTL を適当に設定する。

6. ./opt で実行する。または、SGE などのキューイングシステムが動作する場合には、qsub qsub としてジョブをサブミットする。ここで、最初の qsub はジョブサブミット用のコマンドであるが、次の qsub は make したときに *destination* 内に作られたスクリプトを指定していることになる。

8　入出力について

基本的には、INPUT_DATA/CORD, makefile を設定して make し、実行ディレクトリに移動して*.CNTL を設定すればプログラムの実行準備は終わる。以下に、これらのファイルの設定の仕方を示す。

8.1　CORD

INPUT_DATA/CORD の入力形式は以下の通りである。この例はダイヤモンド構造の Si の場合の例である。

```
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ <<<<<<<<<<<<<<<<<<  Basic Inputs  >>>>>>>>>>>>>>>>>>>>>>>>>>+
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#Atom coordinates relative to unit vectors,       Atom,mode.
 0.250000000D+00  0.250000000D+00  0.250000000D+00  Si o
 0.500000000D+00  0.500000000D+00  0.500000000D+00  Si o
#^              ^  ^              ^  ^              ^  ^^ ^
```

この部分が原子の相対座標（ユニット・ベクトルで与えられる座標に対する位置）を与える。#から#までが入力フィールドであり、この間の行数が原子数を与える。各座標の相対座標は^から^までの桁で数値を与える。また、原子の種類は大文字、小文字を含めて判別しているため、Si のように記載する。mode は原子の動きについて示しており、o は原子位置の最適化を行い、f は原子を固定することを意味する。各元素の荷電子数や擬ポテンシャルの l_0 の情報などは、INPUT_DATA/TABLE_OF_ELEMENTS にある。

```
================<< Other basic inputs >>=========================


 ------------------------------------------------------------------
 | unit_vec:  Primitive vectors of unit cell in real space [Ang]|
 | kp_file:   Sampling type of primitive F.B.Z.                 |
 |                      [file name with 6 char.]                |
 | E_cut:     Cutoff energy for plane-wave [Ry]                 |
 | band_calc: Choice [yes|no|metal|bands|mband]                 |
 | f_crt:     Critical force [Ry/a.u.] for judging convergency. |
 | f_tol:     Parameter for judging convergency of electronic   |
 |            deg. freedom.(Relative )                          |
 | calc_mode: Calculation mode.(bas[ic] or ext[ended])          |
 | ideb:      ideb = 1; Running verbose mode.                   |
 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


 &basic_input
 unit_vec = 2.715, 0.000, 2.715,
            2.715, 2.715, 0.000,
            0.000, 2.715, 2.715,
 kp_file   = 'DP10GM',
 E_cut     = 8.0,
 band_calc = 'metal',
 f_crt     = 5.0D-03,
 f_tol     = 1.0D-8,
 calc_mode = 'bas',
 ideb      = 0
 &end
```

このセクションは、Fortran の namelist 形式で与えられている。

- unit_vec：単位胞を与えるユニット・ベクトルを

$$\begin{matrix} a_{1x} & a_{1y} & a_{1z} \\ a_{2x} & a_{2y} & a_{2z} \\ a_{3x} & a_{3y} & a_{3z} \end{matrix}$$

の順序で与える。このとき、プログラム中での逆格子を与えるベクトルは、

$$\mathbf{a}_i = (a_{ix}, a_{iy}, a_{iz}), \tag{37}$$

として、

$$\mathbf{b}_1 = \frac{\mathbf{a}_2 \times \mathbf{a}_3}{2\pi \mathbf{a}_1 \cdot \mathbf{a}_2 \times \mathbf{a}_3}, \mathbf{b}_2 = \frac{\mathbf{a}_3 \times \mathbf{a}_1}{2\pi \mathbf{a}_1 \cdot \mathbf{a}_2 \times \mathbf{a}_3}, \mathbf{b}_3 = \frac{\mathbf{a}_1 \times \mathbf{a}_2}{2\pi \mathbf{a}_1 \cdot \mathbf{a}_2 \times \mathbf{a}_3}, \tag{38}$$

で与えられる。

- kp_file：k 点のリストを与えるファイルの指定。本パッケージでは結晶系によって k 点リストを取り替えて使っている。'DP10GM' は O 群の対称性のみを考慮した立方晶系用のもの。
- E_cut はカットオフエネルギー E_{cut} を Ry 単位で与える。
- band_calc は計算法のスイッチ。'metal' とした場合にはフェルミ準位近傍でフェルミ関数により準位の占有数をぼかした計算を行うことを意味している。'bands' とした場合には、以下の説明にある'bas' モードを使ってバンド計算をすることができる。
- f_crt：原子間力についての収束判定条件。
- f_tol：電子系の CG に関する収束判定条件。
- calc_mode： 計算モード。単位胞固定で原子位置の最適化までを行う場合には'bas' として使う。単位胞の最適化まで行う場合には'ext' とすることで圧力一定の最適化を行うことが出来る。
- ideb：標準出力をコントロールするパラメータ。

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+<<<<<<<<<<<<<< Mode dependant parameters >>>>>>>>>>>>>>>>>>>>+
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

===============<< Case 1;  band = yes or metal. >>=============


 ----------------------------------------------------------------
 | pwidth: Width of graduall cutoff for occupation number[Ry].|
 | incw:   Eigen states is sorted in every incw CG steps.     |
 | ncband: Number of un-occupied states.                      |
 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


 &METALD
 pwidth=0.005,
```

```
 incw=10,
 ncband=4
 &END
```

Case 1 の部分は、'metal' のときに必要となるパラメータ。pwidth はフェルミ分布関数の幅を与える。incw は準位の入れ替えを行うステップ回数。ncband は計算するフェルミ準位以上にある空の軌道の数。

```
====<<  Case 2;  cmode = ext. (Variable cell simulation.) >>====

 ___[ Constant-cutoff is mimiced with following parameters. ]___
 |                                                              |
 | vke:    Kinetic energy is scaled in vke where                |
 |         |k+G|^2 is beyond E_cut.                             |
 | sigma:  Above scaling is faded in over sigma.                |
 | defmax: Maximum deformation of unit-cell allowed in this     |
 |         simulation.                                          |
 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Original
 vke = 2.0,
 sigma = 0.05,
 defmax = 3*0.05

 &mimicd
 vke = 2.0,
 sigma = 0.10,
 defmax = 3*0.05
 &end
```

Case 2 の部分は、圧力一定の分子動力学計算において単位胞が変化してもカットオフエネルギーを止めている場合に生じうる G 点数の変化を抑える機能に関する部分。本パッケージでは使われていない。

8.2 SOPT.CNTL

Sopt は計算上設定されるセル（計算上の単位胞のことで、結晶を計算している場合でも必ずしも結晶単位胞と一致させる必要はない）を固定したモードでの計算を行う。

Sopt の動作を指定するコントロールファイルである INPUT_DATA/SOPT.CNTL は、Fortran の namelist 形式を用いて書かれている。その入力形式は、以下の通りである。

```
&soptcntl
itmaxs=1,
f_crt=0.005,
ftol = 1.0d-9,
itmaxd=200,
delta = 0.4,
ideb = 0
&end
```

- itmaxs：構造最適化の最大ステップ数。
- f_crt：構造最適化における収束判定条件。各原子に働く力の絶対値の最大値が f_crt 以下になったときに収束したとみなす。単位は A.U.
- ftol：電子系の収束判定条件。1CG ステップあたりのエネルギーの相対変化を単位とする。
- itmaxd：電子系最適化の最大 CG ステップ数。
- delta：原子配置を最適化する時のステップ幅。原子に働く力にかかる比例定数。

8.3 ALLOPT.CNTL

Allopt では計算上設定されるセルも含めて、構造の最適化を行ったり、ある種の分子動力学法で動かすことができる。

INPUT_DATA/ALLOPT.CNTL の入力形式は以下の通りである。

```
0. Chose a mode.

mode = 'md|op': Molecular Dynamics or Optimization.
In case that MD is chosen, you decide whether "Nose's constant
temparature formula" is used or not.

&modesw
mode='md',
nose = 'yes',
&end
```

- Allopt 機能を使う場合には、分子間力と内部応力を用いて mode='md' として分子動力学法として動作させる、または mode='op' として構造最適化を行わせることができる。nose='yes' とすると能勢の方法による温度コントロールを行う。nose='no' のときには温度コントロールなしでの分子動力学を行う。

```
0-a. Mode dependant parameters.

 nose:          In case that nose = 'yes', constant temparature simulation
                with following parameters is performed.
        unit:   The unit of temparature.[au|K|eV]
        temp:   Temperature.
        q_inv:  This value corresponds to Q^(-1).
                In case that q_inv = 0, then zeta = 0.
                See, Nose, Solid State Physics, Vol. 24, pp. 98-106(1989).
                     Nose, Mol. Phys., Vol. 52, p 255(1984).
                 q_inv = 0.05??
 &noseparam
 unit = 'K',
 temp = 100.0,
 q_inv = 0.01,
 &end

 p_crt: Convergence parameter to judge convergence of pressure in [GPa].
 gamma: Friction ratio. gamma is in [0-1]. If gamma =1, MD is done.

 &optparam
 f_crt=1d-10,
 ftol=1d-8,
 p_crt=0.0001,
 gamma = 0.5,
 &end
```

- unit で温度の単位を原子単位、ケルビンまたはエレクトロン・ボルトで指定できる。数値は temp で指定する。また、q_inv は能勢のパラメータ。
- f_crt は構造最適化時における原子間力の収束判定条件。単位は原子単位。
- ftol は電子状態に関するエネルギー収束判定条件。
- p_crt は構造最適化時における圧力の収束判定条件。単位は [Ry]。
- gamma は原子と単位胞の両方に関する速度に対して与えられる減衰定数。

```
================================================================================
1. Define external environment.

--------------------------------------------------------------------------------
```

```
1-1. Pressure

 iso: A switch to select isotropic or anisotropic pressure.
        Case; iso = 'no', pext_3d_[i|f](1:3) is read.
        Case; iso = 'yes', pext_3d_[i|f] is not read.
 pext_ini: initial pressure
 pext_fin: final pressure

 &extpress
 iso='no',
 pext_init=0.0,
 pext_final=0.0,
 pext_3d_i=0.0,0.0,0.0,
 pext_3d_f=0.0,0.0,0.0,
 &end
```

- iso により等方的圧力を指定するか否かを決める。
- pext_init, pext_final または pext_3d_i, pext_3d_f により、分子動力学のプロセスにおいて外部圧力の初期値と終状態での値を指定する。

```
------------------------------------------------------------------------------
1-2. Parameters concernig how the pressure is introduced and how MD runs.

 nsec_p:        External pressure is gradually enforced.
                Sectioned by nsec_p steps between pext_fin and pext_ini.
 nstp_i:        nstp_i MD steps are performed at the pext_ini.
 nstp_g:        nstp_g MD steps are performed at each pressure
                except for pext_ini.

 &prscntl
 nsec_p=10,
 nstp_i=10,
 nstp_g=1,
 &end
```

- nsec_p, nstp_i, nstp_g により外部圧力の設定値の変化を指定する。

```
==============================================================================
3. Parameters for controlling evolution.
```

```
 delt_t:         Time interval in femto second.
 ww:             Fictitious mass ratio [w/mass]
 itmaxa:         Maximam iterations for MD.
 itmaxd:         Maximam iterations in diagw.

 &mdparams
 delt_t=5.0,
 ww = 100000.0,
 itmaxa=500,
 itmaxd=200,
 &end
```

- delt_t：分子動力学シミュレーションにおける時間刻み。
- ww：単位胞に対する仮想的な質量。
- itmaxa：分子動力学計算のステップ総数。
- itmaxd：CD の最適化における最大 CG ステップ数。

```
================================================================================
4. Initial condition.

 ekin_init:      Initial kinetic energy is scaled in this value.
 init_conf:      Switch how is the initial velocities are given.
        Case; init_conf = 'given', read from iveloc namelist in this file.
        Case; init_conf = 'rando', random vectors, generated with 'iseed',
                         is used.
 ivpoint:        Initial kinetic energy is given at ivpoint step.

 &initveloc
 ekin_init = 10.0,
 ckin_init = 0.0,
 init_conf = 'given',
 iseed = 180,
 ivpoint = 1,
 &end

--------------------------------------------------------------------------------
4-1. Define initial direction to move.
```

```
     This parameters are read only in case init_conf = 'given'.

how:  Case; how = 'rel', final conf - initial conf is used.
      Case; how = 'abs', final conf. is used as a vector.

latv_i(3,3): initial primitive vector in real space [Ang]
latv_f(3,3): final primitive vector in real space [Ang]

coordi(3,natm): initial coordinate
coordf(3,natm): final coordinate

Example for structural transformation from cubic-diamond to beta-tin
of silicon.

 &iveloc
 how = 'rel',
 latv_i =
 2.715, 0.000, 2.715,
 2.715, 2.715, 0.000,
 0.000, 2.715, 2.715,
 latv_f =
 2.715, 0.000, 2.715,
 2.715, 2.715, 0.000,
 0.000, 2.715, 2.715,
 coordi =
 0.000000000,  0.000000000,  0.000000000,
 0.500000000,  0.500000000,  0.000000000,
 coordf =
 0.000000000,  0.000000000,  0.000000000,
 0.500000000,  0.500000000,  0.000000000,
 &end
```

- この部分は、初期速度の与え方に関する部分。ESopt では最後の coordi, coordf は読み込まれない。

8.4 FILE_IO.CNTL

INPUT_DATA/FILE_IO.CNTL の入力形式は以下の通りである。

```
=====================================================================
1. Parameter(s) to controll output.
```

```
 &file_io_cntl
 read_file='no',
 initial_file=1
 &end
```

intchg: interval of charge output

```
 &outcntl
 intchg = 10,
 intefl = 20,
 intwf = 10,
 &end
```

2. Debug Flags

```
 &debug_fl
 ideb = 0
 &end
```

- read_file : 計算開始時の原子配置および波動関数として以前の計算結果を利用するかどうかを与える。'cont', 'wyes', 'cyes', 'syes' または'no' が指定できる。それぞれ以下のような動作をする。
  - 'cont': STRUCT/strct から構造を、波動関数を実行ディレクトリーにある fort.98 から読み込む。
  - 'wyes': 波動関数を実行ディレクトリーにある fort.98 から読み込む。
  - 'cyes': STRUCT/cell-XXX および STRUCT/latcrd-XXX を読む。XXX は I3.3 によって与えられる数で、initial_file によって与える。波動関数を実行ディレクトリーにある fort.98 から読み込む。
  - 'syes': STRUCT/cell-XXX および STRUCT/latcrd-XXX を読む。XXX は I3.3 によって与えられる数で、initial_file によって与える。
  - 'no': 構造は CORD で与えられるものを利用し、波動関数は乱数により発生させたものを使う。
- initial_file : read_file='cyes', read_file='syes' のときの XXX の値。read_file がそれ以外のときには無視。

- intchg：電荷密度分布を出力するタイムステップの間隔を与える。
- intefl：MONIT/energy を出力する CG ステップの間隔を指定する。IO 時間を短縮したいときには大きめの値を指定する。
- intwf：MONIT/wave000 を出力する間隔を指定していた。現バージョンでは波動関数の出力は実行ディレクトリーに fort.99 として出力される。計算を接続するときにはこのファイルを fort.98 としてコピーして使う。

8.5 kpoint

kpoint の入力形式は以下の通りである。

```
number of line (i4) =   7
                      ^^^^
 G X K G L K W X
number of mesh (i4) =  10
 1.000000000  0.000000000  0.000000000
 1.000000000  0.500000000  0.500000000
number of mesh (i4) =  10
 1.000000000  0.500000000  0.500000000
 0.750000000  0.375000000  0.375000000
number of mesh (i4) =  10
 0.750000000  0.375000000  0.375000000
 0.000000000  0.000000000  0.000000000
number of mesh (i4) =  10
 0.000000000  0.000000000  0.000000000
 0.500000000  0.500000000  0.500000000
number of mesh (i4) =  10
 0.500000000  0.500000000  0.500000000
 0.750000000  0.375000000  0.375000000
number of mesh (i4) =  10
 0.750000000  0.375000000  0.375000000
 0.750000000  0.250000000  0.500000000
number of mesh (i4) =  10
 0.750000000  0.250000000  0.500000000
 0.500000000  0.000000000  0.500000000
```

ここで、number of line はブリュアン域において何本の k 線上で計算バンド分散を求めるのかを指定する。この数は 20 以内に制限されている。上記のようにこの数が 7 であるとする。この k 線は連続していて一筆書きできるものとする。一行空いて、次の行にはこの 7 本の k 線の両端点における名称を書くことができる。空白

で区切られた文字列を指定することができる。対称性によって決まった名称があればそれを指定してよい。次に、ひとつの k 線ごとに 3 行を用いて、k 線上の分割点数と、両端点を逆格子ベクトル $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{b}_3$ の与える空間内の点として与える。

8.6 出力について

計算の途中経過には、実行ディレクトリにつくられる MONIT というディレクトリにエネルギー等の情報が出力される。全エネルギーは、energy というファイルに出力される。波動関数は、intwf ごとに wave000 という名前で出力される。(ESopt ではこのオプションは停止している。) 構造最適化も終了した時には wave999 という名前で出力される。波動関数は unformatted で出力される。

実空間の電荷密度分布は、CHARGE というディレクトリの下に、chgdnsXXX というファイル名で保存される。

結果は、以下の場所に書き出される。

- Sopt を用いた場合：RESULTS
  - cord*** : 原子座標（実空間での値）
  - eig*** : Kohn-Sham 方程式の固有値
  - gradient*** : 原子間力
  - H*** : 格子ベクトルが作る 3 × 3 行列の対角項、全エネルギー、単位胞の体積
  - Htot*** : 全エネルギーを与える以下のエネルギーが順番に出力されている。１．電子系の運動エネルギー、２．ハートレー項、３．交換・相関項、４．擬ポテンシャルの局所成分によるポテンシャルエネルギー、５．擬ポテンシャルの非局所成分によるポテンシャルエネルギー、６．α 項、７．エバルト項、８．１～７の合計。
- Allopt を用いた場合：STRUCT
  - cell*** :格子ベクトル
  - latcrd*** :内部座標
  - press*** :外部圧力
  - realcrd***.xyz:原子座標
  - result*** :全エネルギー（Sopt の Htot***と同じ)、エネルギーとエンタルピー、圧力、原子間力の順に出力される。
  - strct*** :単位胞のパラメータ, cell***, latcrd, と zeta の値などが出力される。
  - strn*** :歪みテンソルが出力される。

9　プログラム・ソースコード

この節では、各サブルーチンにおいて物理量の計算がどのように実現されているかの例を示すことで、プログラムの動作を理解するための情報を提供する。

9.1　運動エネルギー

運動エネルギーは、etotal.f90 に含まれる関数 ekin(inw) の中で以下のように計算されている。

```
!***********************************************************************
      double precision function ekin(inw)

      use parameters
      use globalarray

      implicit none

      include 'commons/common.h'

      integer :: ik,ib,ipw,inw
      real(8)  frm,z,fd,x

!  -----<< definition of fermi distribution function >>-----

      frm(z) = 1d0 + vke_mim/( 1d0 + exp( -1d0*beta_mim*z ) )

!  ----------<< calc. of KE (mimicing) >>----------
!* epw[Ry],gk[a.u.],nadd

      ekin = 0d0

      do ib = 1,mxband_g
         do ik = 1,nk_g
            fd = fdist_g(ib,ik)
            do  ipw = 1,mxpwk(ik)

               x = 0.5d0*( gk_g(ipw,ik) - ec_mim )

               ekin = ekin &
                    + gk_g(ipw,ik)*frm(x) &
                    * (  dreal(wvfn_g(ipw,ib,ik,inw))**2 &
                    + dimag(wvfn_g(ipw,ib,ik,inw))**2 ) * fd
            end do
         end do
      end do
```

```
      ekin = 2d0*ekin/dfloat(natm_g)/effnk

      return
      end
```

ここで、

- inw: 波動関数は CG ステップを実現するためにステップの経路上で幾つか保存されている。その波動関数を指定するインデックス。
- ib, ik, ipw: それぞれバンドインデックス、$\mathbf{k}$、及び波数ごとに決められる E_{cut} 以下のエネルギーをもつ平面波 $\exp(i(\mathbf{k}+\mathbf{G})\cdot\mathbf{r}))$ を定める $\mathbf{G}$ を示す。
- gk_g(ipw,ik): $|\mathbf{k}+\mathbf{G}|^2$ が格納されている配列。
- wvfn_g(ipw,ib,ik,inw): 波動関数 $\phi_{\mathbf{k},n}(\mathbf{G})$ を格納した配列。

である。よって、変数 ekin に

$$E_{kin} = 2\sum_{\mathbf{k},n,\mathbf{G}} |\mathbf{k}+\mathbf{G}|^2 |\phi_{\mathbf{k},n}(\mathbf{G})|^2 ,$$

が代入される。これは、ekin は単位体積当たりのエネルギーを、単位を Rydberg 単位 (電子質量 $m = 1/2$ とする) として、スピン自由度の和からくる因子 2 を含めて与えているためである。さらに、因子 frm(x) によって、単位胞の変形 (ESopt では圧力一定シミュレーションの機能を止めてあるが) を許した際に、実効的にエネルギーカットオフ一定の条件が満たされるようにしている。

9.2 ハートレー・エネルギー

ハートレー・エネルギーは、etotal.f90 に含まれる関数 eee(vcell) の中で以下のように計算されている。

```
!***************************************************************************
      double precision function eee(vcell)

      use parameters
      use globalarray

      implicit none

      include 'commons/common.h'

      integer :: ig
      real(8) :: vcell
```

```
      eee = 0d0

      do ig = 2,ng_g

         eee = eee + (dreal(chgg_g(ig))**2+dimag(chgg_g(ig))**2)*gg_g(ig)

      end do

      eee = eee*4d0*pi/vcell/dfloat(natm_g)

      return
      end
```

ここで、

- chgg_g(ig): 密度のフーリエ変換 $n(\mathbf{G})$ を格納した配列。
- gg_g(ig): $1/|\mathbf{G}|^2$ が格納されている配列。

である。よって、変数 eee に

$$E_{el-el} = \frac{4\pi}{\Omega_{cell}} \sum_{\mathbf{G}}{}' \frac{|n(\mathbf{G})|^2}{|\mathbf{G}|^2} ,$$

が代入される。これは、eee も単位体積当たりのエネルギーを、単位を Rydberg 単位 (電子質量 $e^2 = 2$ とする) として与えているためである。Ω_{cell} でむしろ割られているのは、本来 E_{el-el} を与えていた実空間積分が 2 重積分になっていることから来ている。

9.3 全エネルギー

全エネルギーは、etotal.f90 に含まれるサブルーチン etotal(vcell,inw,etot) の中で以下のように計算されている。

```
!***********************************************************************
      subroutine etotal (vcell,inw, etot)

      use parameters
      use globalarray

      implicit none

      integer :: lmx,inw,i,j
      real(8) :: etot(8),vcell,vatm, &
           ekin,eee,exc,evl,evnl,t1,t2,dsecnd
```

```
parameter( lmx = 2 )

include 'commons/common.h'

t1 = dsecnd('cpu')

vatm = vcell/dfloat(natm_g)

etot(1) = ekin(inw)
etot(2) = eee(vcell)
etot(3) = exc(vcell)
etot(4) = evl()
etot(5) = evnl(inw)

etot(8) = etot(1)+etot(2)+etot(3)+etot(4)+etot(5)+etot(6)+etot(7)

t2 = dsecnd('cpu')

cpu_time(1) = cpu_time(1) + t2-t1

return
end
```

エネルギーは項ごとに計算され、それぞれ以下の値が入る。

- ' ekin = ',etot(1) : 運動エネルギー
- ' eee = ',etot(2) : ハートレー・エネルギー
- ' exc = ',etot(3) : 交換相関エネルギー
- ' evl = ',etot(4) : 擬ポテンシャルの局所項の寄与
- ' evnl = ',etot(5) : 擬ポテンシャルの非局所項の寄与
- ' ealp = ',etot(6) : α_I の寄与
- ' eewd = ',etot(7) : イオン-イオン間相互作用 (エバルト和)
- ' etot = ',etot(8) : 全エネルギー

9.4 Sopt コントロール・ルーチン

main ルーチンはもともと幾つかのモードに対して分岐する形となっていた。

- Allopt : 格子構造も含めた最適化、圧力一定の分子動力学などを行うモード。
- Sopt : 格子を止めたままで、原子位置の最適化を行うモード。

- DMD : 減衰効果を含む分子動力学モード。

ESopt は、Sopt, Allopt の一部の機能以外を停止した形で提供している。そこで、Sopt の本体であるサブルーチン sopt を次に考察する。以下では実際のものから無駄な出力等は省いてある。

```
!***********************************************************************
      subroutine sopt()

      use parameters
      use globalarray

      implicit real*8(a-h,o-z)

      include 'commons/lattice.h'
      include 'commons/common.h'

      logical(4) ::  exist1, exist2
      real(8) :: etot(8),delta,cpu0,cpu1

      namelist / soptcntl / itmaxs, ftol, f_crt, itmaxd, delta, ideb

!=======================================================================

      inquire(file='SOPT.CNTL',exist=exist2)
      inquire(file='../INPUT_DATA/SOPT.CNTL',exist=exist1)

      if(exist2) then
         open(10,file='SOPT.CNTL',form='formatted', &
              iostat=ios,status='old')
         read(10,soptcntl,iostat=ios)
         close(10)
      else
         if(exist1) then
            open(10,file='../INPUT_DATA/SOPT.CNTL',form='formatted',&
                 iostat=ios,status='old')
            read(10,soptcntl,iostat=ios)
            close(10)
         else
            stop 'Can not open SOPT.CNTL'
         endif
      endif

!  ----------< generate g-points >---------------

      call gptgen()

!  -----< calc. some components of total energy >----------
```

```
      call vinit()

!  -----< calc. ewald sum (independent part with coordinates of atoms ) >-----

      call  iniewd()

! Main loop starts.
      do its = 1, itmaxs

! ----------< optimization for electronic state >----------

      call getll(etot,nloop)

      call charge(0)

! ----------< calc. Hellmann-Feynman force >----------

      call force(vcell)

      call eforce(natm_g,stoa,grd,frc,fn)

      call wfile(etot,its)

! ----------< output data >----------

      call wfile0(000,its)

! ----------<  judgement of convergence >-------------

      if(ideb.eq.1) write(*,*)'fn, f_crt = ',fn,f_crt
      if(fn.lt.f_crt) exit

! ----------< move all atoms >--------------------------------

      call mvatm(delta)

      end do
! Main loop ends.

      if(fn.lt.f_crt) then
         call wfile0(999,its)
      end if

      write(*,*)'elapse time in sopt = ', cpu1-cpu0
      return

      end subroutine sopt
```

これにより、アルゴリズムの流れになっていることが分かる。

1. 初期化部分
   (a) SOPT.CNTL 読み込み
   (b) gptgen() : **G** などのデータを作成。
   (c) vinit() : 擬ポテンシャル・データの入力。
   (d) iniewd() : エバルト和計算のための初期化。
2. 構造最適化ループ
   (a) getll(etot,nloop) : 電子状態決定用ルーチン
   (b) charge(0) :電荷密度分布計算ルーチン
   (c) force(vcell) : 原子間力計算ルーチン
   (d) eforce(natm_g,stoa,grd,frc,fn) : 最適化の方法により force を調整するルーチン
   (e) wfile(etot,its), wfile0(999,its), wfile0(000,its) : 出力用ルーチン群
   (f) mvatm(delta) : 原子の移動を起こさせるルーチン

9.5 対角化用サブルーチン

サブルーチン diagw において、電子状態の CG 法による最適化が行われる。

```
subroutine diagw (nloop,vcell,alp,etot)

use parameters
use globalarray

implicit none

include 'commons/common.h'

integer :: nloop, inw, itmax, ik, in, ig, ist, loop, its, nwf, nnn, &
     istc, i

real(8) ::  etot(8), vcell, alp, eps, ang, d1, d2, dnf, step, ep, e_p,&
     fret, fp, gd, avm, dmm, t1, t2, dsecnd,e2,e1

complex(8) :: ggc,gam,s, chg1

parameter (eps = 1d-10)
```

```
      parameter (inw = 2)
      parameter (e_p = 5d0)

! -----< calc. charge density for G-space >-----
      call schmit(inw)

      call toreal(inw)

      chg1 = chgg_g(1)
      etot(6)=alp*dreal(chg1)

! -----< calc. total energy(fp) >-----

      call etotal (vcell,inw,etot)

      fp = etot(8)

! -----< calc. force >-----

      call wforce(vcell,inw)

      call precnd(e_p)

      call schmi2(inw,1)

      itmax = max(itmaxd*2,npw_g*mxband_g*nk_g*4)

! nwf(= npw*nband*nk) is num. of wave func.

! -----< store old data of w.f. >-----

      do in = 1, mxband_g
         do ik = 1, nk_g
            do ig = 1, mxpwk(ik)
               wvfn_g(ig,in,ik,4)=wvfn_g(ig,in,ik,1)
               wvfn_g(ig,in,ik,5)=wvfn_g(ig,in,ik,1)
            end do
         end do
      end do

! -----< iteration start >-----

      ist = 1
      istc = 0
      loop = 0
! Main loop starts
      do its = 1, itmax

      s = (0d0,0d0)
```

```
      d1= 0d0
      d2= 0d0

! -----< summation of some value >-----
      nwf = mxband_g*nk_g

      do in = 1,mxband_g
         do ik = 1,nk_g
            do ig = 1,mxpwk(ik)

               s = s + dconjg(wvfn_g(ig,in,ik,1))*wvfn_g(ig,in,ik,2)
               d1= d1+dreal(wvfn_g(ig,in,ik,1))**2+dimag(wvfn_g(ig,in,ik,1))**2
               d2= d2+dreal(wvfn_g(ig,in,ik,2))**2+dimag(wvfn_g(ig,in,ik,2))**2

            end do
         end do
      end do

      ang = acos(cdabs(s)/sqrt(d1*d2))/pi
         dnf = sqrt( d1/nwf )

      if(ang.lt.0.46d0.and.ang.ge.0.385d0) then
         call schmi2(inw,2)
      elseif(ang.lt.0.385d0)then
         call schmi2(inw,1)
      endif

      call linmin(vcell,etot,fret,ist)

      if(ist.eq.0) istc=istc+1

      step=2d0*dabs(fret-fp)/(dabs(fret)+dabs(fp)+eps)

!  ----------< judgement for convergency >----------

      if((step.le.ftol.and.ist.eq.1).or.its.gt.itmaxd.or.istc.gt.3) then

       loop = loop + 1
       if(loop.gt.2) then

          etot(6)=alp*dreal(chg1)
          call efile(etot,ang,dnf,its,ftol,step,loop)
          nloop = its
          write(*,*) 'Count (its,loop)', its,loop
         return

       else

          ist = 0
```

```
       endif
      endif

!  ------------------------------------------------

      ep=fret-fp

      fp = fret
      call wforce(vcell,inw)
      call precnd(e_p)

      call eigenv(avm,dmm,its,nnn)

      if(ist.eq.0.or.nnn.eq.1) then
      call toreal(inw)
      chg1 = chgg_g(1)
      call wforce(vcell,inw)
      call precnd(e_p)
      endif

      if(its.eq.0) ist = 0
      if(ist.eq.1) then
         gd = 0d0
         ggc = (0d0,0d0)

         do in = 1, mxband_g
            do ik = 1, nk_g
               do ig = 1, mxpwk(ik)
                  gd = gd + dreal(wvfn_g(ig,in,ik,4))**2 &
                       + dimag(wvfn_g(ig,in,ik,4))**2
                  ggc = ggc + dconjg(wvfn_g(ig,in,ik,1) &
                       - wvfn_g(ig,in,ik,4))*wvfn_g(ig,in,ik,1)
               end do
            end do
         end do

         if(gd.eq.0d0) then
            write(*,*)'warning. really gd = 0 ?'
            return
         endif

         gam = ggc/gd

         do in = 1, mxband_g
            do ik = 1, nk_g
               do ig = 1, mxpwk(ik)
                  wvfn_g(ig,in,ik,4)=wvfn_g(ig,in,ik,1)
               end do
```

```
            end do
         end do

         do in = 1, mxband_g
            do ik = 1, nk_g
               do ig = 1, mxpwk(ik)
                  wvfn_g(ig,in,ik,5)=wvfn_g(ig,in,ik,4) &
                       + gam*wvfn_g(ig,in,ik,5)
               end do
            end do
         end do

         do in = 1, mxband_g
            do ik = 1, nk_g
               do ig = 1, mxpwk(ik)
                  wvfn_g(ig,in,ik,1)=wvfn_g(ig,in,ik,5)
               end do
            end do
         end do

      elseif(ist.eq.0) then

         do in = 1,mxband_g
            do ik = 1,nk_g
               do ig = 1,mxpwk(ik)
                  wvfn_g(ig,in,ik,4)=wvfn_g(ig,in,ik,1)
                  wvfn_g(ig,in,ik,5)=wvfn_g(ig,in,ik,1)
               end do
            end do
         end do

         ist = 1

      endif

      nnn = mod(its,intefl)

      if(nnn.eq.0) then
         if(ideb.eq.1) write(*,*) 'efile in'
         call efile(etot,ang,dnf,its,ftol,step,loop)
         if(ideb.eq.1) write(*,*) 'efile out'
      endif

      end do
! Main loop ends

      stop 'Maximum iterations exceeded (diagw).'

      end subroutine diagw
```