

第一原理電子状態計算ソフトウェア
PHASE/0 2019
ユーザーマニュアル

PHASE/0



PHASE System

COPYRIGHT of the program codes

Copyright (C) of the original version: Hideki Katagiri, Koichi Kato, Tsuyoshi Miyazaki, Yoshitada Morikawa, Hideaki Sawada, Toshihiro Uchiyama, Tsuyoshi Uda, Takahiro Yamasaki.

Copyright (C) of the developed version by the national projects FSIS, RSS21, and RISS has been managed by the Institute of Industrial Science (IIS), the University of Tokyo.

The Institute of Industrial Science (IIS) has a right to distribute the program set developed from the original version as a free software.

HISTORY

The original version of this set of the computer programs "PHASE" was developed by the members of the Theory Group of Joint Research Center for Atom Technology (JRCAT), based in Tsukuba, in the period 1993-2001. The names of the contributors to the original version are Hideki Katagiri, K. Kato, T. Miyazaki, Y. Morikawa, H. Sawada, T. Uchiyama, T. Uda and T. Yamasaki. These contributors has agreed with that the Institute of Industrial Science (IIS), the University of Tokyo, distributes this program as a free software.

Since 2002, this program set had been intensively developed as a part of the following national projects supported by the Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan: "Frontier Simulation Software for Industrial Science (FSIS)" from 2002 to 2005, "Revolutionary Simulation Software (RSS21)" from 2006 to 2008. "Research and Development of Innovative Simulation Software (RISS)" from 2008 to 2013. These projects is lead by the Center for Research on Innovative Simulation Software (CISS), the Institute of Industrial Science (IIS), the University of Tokyo.

Since 2013, this program set has been further developed centering on PHASE System Consortium.

The activity of development of this program set has been supervised by Takahisa Ohno.

CONTACT ADDRESS

PHASE System Consortium

E-mail: phase_system@nims.go.jp URL <https://azuma.nims.go.jp>

* When distributing the software "PHASE" duplications, the user must attach the full text in this file.

目次

1. はじめに.....	14
1.1 PHASE システムの概要.....	14
1.2 PHASE とは?	15
1.2.1 PHASE の主な特徴、機能 (PHASE で何が計算できるか?)	15
1.2.2 PHASE/0 のプログラム構成.....	16
1.2.3 利用可能な動作環境 (計算機環境要件)	16
1.2.4 PHASE/0 の機能 (2D 並列版と 3D 並列版の比較)	17
1.3 マニュアルの構成.....	18
1.4 PHASE の更新履歴.....	19
2. PHASE/0 の基本的な利用方法.....	22
2.1 PHASE/0 の計算手順の概要.....	22
2.2 入力データの準備.....	23
2.2.1 計算に最低限必要なファイル.....	23
2.2.2 入力パラメータファイル: <i>nfnp.data</i> (簡略版)	24
2.2.2.1 入力パラメータファイル例.....	24
2.2.2.2 Control ブロック	26
2.2.2.3 Accuracy ブロック.....	26
2.2.2.4 Structure ブロック.....	27
2.2.2.5 Wavefunction_solver ブロック	27
2.2.2.6 Charge_mixing ブロック	28
2.2.2.7 Postprocessing ブロック.....	28
2.2.2.8 入力パラメータファイルの最低限の設定項目	29
2.2.3 擬ポテンシャルファイル.....	32
2.2.3.1 擬ポテンシャルの種類 (内殻の取り扱い方法): フローズンコア型と PAW 型.....	32
2.2.3.2 擬ポテンシャルファイルの入手方法.....	32
2.2.3.3 擬ポテンシャルの指定方法.....	32
2.2.4 入出力ファイル設定ファイル <i>file_names.data</i>	33
2.3 計算の実行.....	35
2.3.1 プログラム <i>phase</i> の実行.....	35
2.3.2 並列計算について.....	35
2.3.2.1 基本的な並列計算.....	35
2.3.2.2 ディレクトリ並列による計算	35
2.3.2.3 レプリカ並列による計算.....	36
2.3.2.4 <i>k</i> 点数の調べ方.....	36
2.3.3 計算状況の確認.....	37
2.3.4 継続計算.....	37
2.3.5 <i>ekcal</i> の実行 (状態密度計算、バンド構造計算)	37
2.3.5.1 状態密度の計算.....	37
2.3.5.2 バンド構造計算.....	38
2.3.5.3 3D 版について	39
2.4 計算結果の確認.....	40
2.4.1 計算終了の状況・要因・対応.....	40
2.4.2 計算終了 (正常終了、異常終了) の確認.....	40
2.4.3 SCF 計算、構造緩和計算の収束状況の確認.....	41
2.4.4 計算状況の確認 (ログファイル <i>output000</i> および <i>jobstatus000</i>).....	41
2.4.4.1 サンプリング <i>k</i> 点.....	42

2.4.4.2	全エネルギー	42
2.4.4.3	スピン状態	42
2.4.4.4	固有値および占有数	42
2.4.4.5	SCF 計算あたりの計算時間	43
2.4.4.6	計算の進捗状況 (jobstatus000)	43
2.5	計算結果の解析、可視化	44
2.5.1	全エネルギー、原子に働く力の最大値 (エネルギー履歴ファイル <i>nfefn.data</i>)	44
2.5.1.1	構造緩和計算	44
2.5.1.2	分子動力学法計算	44
2.5.1.3	格子最適化計算	45
2.5.2	原子座標 (原子座標履歴ファイル <i>nfdynm.data</i>)	45
2.5.3	電荷密度 (電荷密度ファイル <i>nfchr.cube</i>)	47
2.5.4	状態密度 (状態密度ファイル <i>dos.data</i>)	47
2.5.5	バンド構造 (固有値データファイル <i>nfenergy.data</i>)	48
2.6	参考文献	49
3.	入力パラメータファイル: NFINP.DATA (詳細版)	50
3.1	入力パラメータファイルの形式(F_INP ファイル)	50
3.1.1	パラメータ設定形式	50
3.1.2	単位の指定	51
3.1.3	コメント	51
3.1.4	入力パラメータファイル例	52
3.2	入力パラメータファイル NFINP.DATA のタグ (キーワード) の一覧	54
3.3	全体的な計算条件設定 (CONTROL)	61
3.4	計算精度の指定 (ACCURACY)	61
3.4.1	カットオフエネルギー	61
3.4.2	バンド数	62
3.4.3	k 点サンプリングとスメアリング	62
3.4.4	交換相関エネルギー	63
3.4.5	収束判定	63
3.4.6	初期波動関数と初期電荷密度	64
3.4.7	実空間法	64
3.5	原子構造 (STRUCTURE)	65
3.5.1	ユニットセル	66
3.5.2	原子座標	66
3.5.3	原子種の指定	67
3.5.4	系全体への電荷の付加	68
3.5.5	磁気モーメントの初期値の指定	68
3.5.6	対称性の指定	69
3.6	波動関数ソルバー (WAVEFUNCTION_SOLVER)	70
3.6.1	PHASE における計算フロー	70
3.6.2	波動関数ソルバー	70
3.7	電荷密度混合法 (CHARGE_MIXING)	72
3.7.1	電荷密度混合法	72
3.7.2	収束を加速させるテクニック	73
3.8	波動関数ソルバーおよび電荷密度混合法の自動設定	75
3.9	構造最適化 (STRUCTURE_EVOLUTION)	77
3.9.1	構造最適化	77
3.9.2	分子動力学法計算	78
3.9.3	構造更新時の電荷密度、波動関数の予測更新 (収束性の向上)	78

3.10 後処理 (POSTPROCESSING)	80
3.10.1 状態密度 (DOS)	80
3.10.2 電荷密度.....	80
3.10.3 構造最適化/分子動力学シミュレーションの最中に後処理を行う方法.....	81
3.11 ログレベル (PRINTLEVEL)	81
4. 基本機能を利用した計算例.....	82
4.1 全エネルギー計算	82
4.1.1 入力パラメータ	82
4.1.2 計算の実行.....	84
4.1.3 計算結果の出力.....	84
4.2 対称性を考慮した計算.....	86
4.2.1 入力パラメータ	86
4.2.1.1 単位胞の指定.....	86
4.2.1.2 対称性の指定.....	88
4.2.1.3 反転対称性がある場合.....	89
4.2.2 計算例：シリコン結晶(Si2).....	91
4.3 スピン分極を考慮した計算.....	96
4.3.1 強磁性の計算.....	96
4.3.1.1 入力パラメータ	96
4.3.1.2 計算結果の出力.....	97
4.3.2 反強磁性の計算.....	98
4.3.2.1 入力パラメータ	98
4.4 構造最適化.....	100
4.4.1 入力パラメータ	100
4.4.2 計算結果の出力.....	101
4.4.3 計算例：シリコン結晶の構造最適化.....	101
4.5 表面の計算.....	104
4.5.1 表面の計算を実行するには.....	104
4.5.2 反転対称性を考慮した表面の計算.....	106
4.5.3 計算例：金属表面の生成エネルギー.....	108
4.6 原子・分子の計算	110
4.6.1 入力パラメータ	110
4.7 電荷密度の出力.....	112
4.8 状態密度の計算	114
4.9 バンド構造の計算	116
4.9.1 k 点のデータの作成.....	116
4.9.2 固定電荷の計算を行う.....	117
4.9.2.1 入力パラメータ	117
4.9.3 バンド構造図の作成.....	119
4.10 格子定数	121
4.10.1 計算方法.....	121
4.10.2 計算例：Si 結晶.....	121
5. 応用機能.....	123
5.1 解析機能.....	123
5.1.1 ストレステンソル	123
5.1.1.1 機能の概要.....	123
5.1.1.2 入力パラメータ	123
5.1.1.3 弾性定数	124
5.1.1.4 ストレステンソルの補正.....	126

5.1.2 局所状態密度と部分電荷密度	128
5.1.2.1 機能の概要	128
5.1.2.2 原子分割局所状態密度	129
5.1.2.3 層分割局所状態密度	130
5.1.2.4 部分電荷密度	132
5.1.2.5 ウルトラスフト型擬ポテンシャルを利用している場合の高速化	135
5.1.3 射影状態密度	137
5.1.3.1 入力パラメータ	137
5.1.3.2 計算結果の出力	138
5.1.3.3 計算例：BaTiO ₃ 結晶の射影状態密度	138
5.1.4 ワニエ関数	141
5.1.4.1 機能の概要	141
5.1.4.2 計算例：Si の最大局在ワニエ関数	141
5.1.4.3 Wannier90 を用いたワニエ関数解析	143
5.1.5 STM 像解析	148
5.1.6 XPS 解析	149
5.1.6.1 機能の概要	149
5.1.6.2 計算の実行方法	152
5.1.6.3 計算例：Si(100)表面	152
5.1.6.4 参考文献	155
5.1.7 仕事関数	156
5.1.7.1 機能の概要	156
5.1.7.2 入力パラメータ	156
5.1.7.3 計算の実行方法	156
5.1.7.4 計算結果の出力	157
5.1.7.5 計算例：アルミニウムの仕事関数	157
5.1.8 陽電子寿命解析	159
5.1.8.1 機能の概要	159
5.1.8.2 入力パラメータ	160
5.1.8.3 計算結果の出力	161
5.1.8.4 使用上の注意	162
5.1.8.5 参考文献	163
5.1.9 ボルン有効電荷	163
5.1.9.1 概要	163
5.1.9.2 ボルン有効電荷の計算の流れ	163
5.1.9.3 入力データ	163
5.1.9.4 berry.pl の実行	167
5.1.9.5 計算結果	169
5.1.9.6 例題	170
5.1.9.7 berry.pl を使わずにボルン有効電荷を計算する方法	171
5.1.10 BoltzTraP を利用した解析(バージョン 2019.01 以上)	172
5.1.10.1 入力ファイル	173
5.1.10.2 PHASE/0 の実行	173
5.1.10.3 BoltzTraP の実行	173
5.1.10.4 計算例	175
5.1.10.5 参考文献	177
5.2 原子ダイナミクス	178
5.2.1 振動解析	178
5.2.1.1 機能の概要	178
5.2.1.2 入力パラメータ	179

5.2.1.3	計算結果の出力	180
5.2.1.4	計算例：水分子の振動解析	181
5.2.1.5	計算例：シリコン結晶(Si2)	185
5.2.2	フォノンバンド計算	188
5.2.2.1	機能の概要	188
5.2.2.2	利用方法	188
5.2.2.3	例題	194
5.2.2.4	使用上の注意	201
5.2.3	分子動力学法シミュレーション	202
5.2.3.1	機能の概要	202
5.2.3.2	入力パラメータ	202
5.2.3.3	計算結果の出力	203
5.2.3.4	使用方法：エネルギー一定の分子動力学シミュレーション	204
5.2.3.5	使用方法：温度一定の分子動力学シミュレーション	206
5.2.3.6	使用方法：温度・圧力一定の分子動力学シミュレーション	208
5.2.3.7	分子動力学シミュレーションにまつわるその他の設定	212
5.2.3.8	使用における注意点	215
5.3	高精度な電子状態の計算	217
5.3.1	DFT+U 法	217
5.3.1.1	機能の概要	217
5.3.1.2	入力パラメータ	218
5.3.1.3	計算結果の出力	219
5.3.1.4	計算例：立方晶 SrTiO ₃	220
5.3.1.5	計算例：立方晶 LaVO ₃	221
5.3.1.6	計算例：斜方晶 LaVO ₃	221
5.3.1.7	立方晶 FeO	221
5.3.2	ハイブリッド汎関数	223
5.3.2.1	機能の概要	223
5.3.2.2	入力パラメータ	223
5.3.2.3	計算例	226
5.3.2.4	使用における注意点	228
5.3.2.5	参考文献	228
5.3.3	SC-DFT 法	230
5.3.3.1	機能の概要	230
5.3.3.2	理論	230
5.3.3.3	使い方	230
5.3.3.4	計算例	231
5.3.4	ファンデルワールス相互作用（非局所相関項）	232
5.3.4.1	機能の概要	232
5.3.4.2	非局所相関項を含めた全エネルギー計算	233
5.3.4.3	計算例 1：積層グラファイトの全エネルギー計算	236
5.3.4.4	計算例 2：積層 2H-MoS ₂ の全エネルギー計算	238
5.3.4.5	制限事項	238
5.3.4.6	参考文献	238
5.3.5	ファンデルワールス相互作用補正機能	239
5.3.5.1	機能の概要	239
5.3.5.2	入力パラメータ	241
5.3.5.3	計算例	243
5.3.6	有効遮蔽体法 (ESM 法)	244
5.3.6.1	機能の概要	244

5.3.6.2	入力パラメータ	244
5.3.6.3	計算の実行.....	245
5.3.6.4	計算例	245
5.3.6.5	使用における注意点.....	247
5.3.6.6	参考文献	247
5.3.6.7	ライセンス.....	247
5.3.7	<i>Dipole</i> 補正.....	250
5.3.7.1	入力パラメーターファイルの書き方	250
5.3.7.2	ログファイルの見方.....	251
5.3.7.3	計算例	252
5.4	化学反応の解析	254
5.4.1	<i>NEB</i> 法.....	254
5.4.1.1	機能の概要.....	254
5.4.1.2	入力パラメータ	255
5.4.1.3	計算の実行方法	263
5.4.1.4	計算結果の出力	263
5.4.1.5	計算例：シリコン表面に水素分子が解離吸着する反応.....	265
5.4.1.6	使用における注意点.....	268
5.4.1.7	参考文献	269
5.4.2	拘束条件付きダイナミクスと <i>Blue Moon</i> 法による自由エネルギー解析.....	270
5.4.2.1	機能の概要.....	270
5.4.2.2	入力パラメータ	270
5.4.2.3	計算の実行方法	275
5.4.2.4	計算結果の出力	275
5.4.2.5	<i>Blue Moon</i> 法による自由エネルギーの計算.....	276
5.4.2.6	計算例：H ₂ O ₂ およびH ₂ S ₂ 分子の回転障壁の解析.....	278
5.4.2.7	使用における注意点.....	279
5.4.2.8	参考文献	279
5.4.3	<i>Meta-dynamics</i> 法.....	280
5.4.3.1	機能の概要.....	280
5.4.3.2	入力パラメータ	281
5.4.3.3	計算の実行方法	286
5.4.3.4	計算結果の出力	287
5.4.3.5	計算例：炭化水素のエネルギー表面	288
5.4.3.6	使用における注意点.....	293
5.4.3.7	参考文献	293
5.5	時間依存密度汎関数法計算 (TDDFT)	294
5.5.1	実時間処理型の時間依存密度汎関数理論 (<i>RTTDDFT</i>) による光学スペクトル計算.....	294
5.5.1.1	機能の概要.....	294
5.5.1.2	入力パラメータ	294
5.5.1.3	計算の実行方法	295
5.5.1.4	計算結果	295
5.5.1.5	例題.....	296
5.5.1.6	使用上の注意.....	297
5.6	構造最適化.....	299
5.6.1	ストレステンソルを利用したユニットセル最適化機能.....	299
5.6.1.1	入力パラメータ	299
5.6.1.2	計算結果の出力	299
5.6.1.3	計算例：ルチル型TiO ₂	301
5.7	ノンコリニア系の計算、スピン軌道相互作用計算.....	303

5.7.1 ノンコリニア系の計算.....	303
5.7.1.1 機能の概要.....	303
5.7.1.2 入力パラメータ	303
5.7.1.3 計算結果の出力	304
5.7.2 スピン軌道相互作用計算.....	304
5.7.2.1 機能の概要.....	304
5.7.2.2 入力パラメータ	304
5.7.2.3 計算結果の出力	304
5.7.2.4 計算例：O ₂ 分子、Pt ₂ 分子.....	305
5.7.2.5 Si 結晶のバンド	305
5.7.2.6 使用上の注意.....	306
5.8 ELNES/XANES 解析機能.....	307
5.8.1 はじめに.....	307
5.8.2 準備：擬ポテンシャル.....	308
5.8.3 計算の流れ.....	308
5.8.4 基底状態 (A) の SCF 計算	309
5.8.4.1 入力.....	309
5.8.4.2 出力.....	309
5.8.5 内殻電子励起状態 (B) の SCF 計算.....	309
5.8.5.1 入力.....	309
5.8.5.2 出力.....	310
5.8.6 内殻電子励起状態 (B) のスペクトル計算.....	310
5.8.6.1 入力.....	310
5.8.6.2 出力.....	311
5.8.7 計算事例.....	312
5.8.7.1 LiF 結晶の Li 原子 K 端のスペクトル.....	312
5.8.8 文献.....	312
5.9 化学ポテンシャル一定のシミュレーション	313
5.9.1 概要.....	313
5.9.2 入力パラメーター	313
5.9.3 計算の実行.....	314
5.9.4 例題.....	314
5.9.5 使用上の注意.....	315
5.9.6 参考文献.....	315
6. PAW 法による計算.....	316
6.1 機能の概要.....	316
6.2 入力パラメータ、計算の実行方法.....	316
6.3 計算例：体心立方構造クロム	316
6.4 PAW 法の精度を向上させる方法	318
6.5 PAW 法で有効な計算機能一覧.....	318
7. 計算精度など補足事項.....	320
7.1 計算精度、収束性.....	320
7.1.1 カットオフエネルギーと計算精度.....	320
7.1.2 k 点サンプリングと計算精度.....	320
7.1.3 収束判定と計算精度.....	321
7.1.4 ベンチマーク計算例（波動関数ソルバーの比較）	323
7.1.4.1 FCC-Cu.....	323
7.1.4.2 Fe(100) 表面.....	326
7.1.5 波動関数ソルバーおよび電荷密度混合法の自動設定	329

7.2 構造最適化.....	330
7.2.1 構造最適化手法.....	330
7.2.1.1 計算例.....	330
7.2.1.2 計算結果.....	331
7.2.2 構造更新時の電荷密度、波動関数の予測更新（収束性の向上）.....	331
7.3 PHASE/0 の単位系.....	333
8. PHASE/0 のインストール.....	334
8.1 動作環境.....	334
8.2 次元版および3次元版について.....	334
8.3 インストール方法.....	335
9. PHASE/0 プログラム、ツールの実行方法.....	336
9.1 プログラム PHASE.....	337
9.1.1 プログラム <i>phase</i> の実行.....	337
9.1.2 プログラム <i>phase</i> の並列計算オプション.....	337
9.1.2.1 バンド並列、k 点並列.....	337
9.1.2.2 レプリカ並列.....	337
9.1.3 プログラム <i>phase</i> 3 次元並列 (G 点並列) 版.....	338
9.1.4 3 次元版のオプションについて.....	338
9.2 プログラム EKCAL.....	339
9.3 プログラム EPSMAIN.....	340
9.4 プログラム TDLRMAIN.....	340
9.5 状態密度図作成ツール DOS.PL.....	341
9.5.1 <i>dos.pl</i> のオプション.....	341
9.6 K 点ファイル生成ツール BAND_KPOINT.PL.....	343
9.7 バンド構造図作成ツール BAND.PL.....	344
9.7.1 <i>band.pl</i> の実行.....	344
9.7.2 <i>band.pl</i> のオプション.....	344
9.8 原子構造の拡張 TRAJECTORY 形式への変換ツール DYNAM2TR2.PL.....	346
9.9 振動数レベル図作成ツール FREQ.PL.....	348
9.9.1 <i>freq.pl</i> のオプション.....	348
9.10 基準振動の軌跡の拡張 TRAJECTORY 形式ファイル変換ツール ANIMATE.PL.....	350
9.11 座標データ変換ツール CONV.PY.....	351
10. 入出力ファイル.....	353
10.1 入力ファイル.....	353
10.1.1 入力パラメータファイル <i>nfnp.data</i>	353
10.1.2 擬ポテンシャルファイル.....	353
10.2 入出力ファイル設定ファイル FILE_NAME.DATA.....	355
10.3 入力ファイル（プログラム EKCAL）.....	355
10.3.1 サンプリング k 点ファイル <i>kpoint.data</i> (<i>F_KPOINT</i>).....	355
10.4 出力ファイル.....	356
10.4.1 状態密度ファイル <i>dos.data</i> (<i>F_DOS</i>).....	356
10.4.2 エネルギー履歴ファイル <i>nfeff.data</i> (<i>F_ENF</i>).....	358
10.4.3 原子座標履歴ファイル <i>nfdynm.data</i> (<i>F_DYNAM</i>).....	360
10.4.4 電荷密度ファイル <i>nfchr.cube</i> (<i>F_CHR</i>).....	361
10.4.5 継続計算用ファイル <i>continue.data</i> (<i>F_CNTN</i>).....	362
10.4.6 固有値データファイル <i>nfenergy.data</i> (<i>F_ENERG</i>).....	363
11. 誘電応答解析機能 UVSOR.....	365

11.1	はじめに	365
11.1.1	UVSOR とは	365
11.1.2	UVSOR の特徴と機能	366
11.1.3	更新履歴	367
11.1.4	パッケージの構成	368
11.2	計算手法	369
11.2.1	電子系	369
11.2.1.1	計算手法[1]	369
11.2.1.2	遷移確率の計算	369
11.2.1.3	電子系誘電率の計算	370
11.2.1.4	光学スペクトルの計算法	371
11.2.1.5	第一原理法による計算法	371
11.2.1.6	Read and Needs 法	371
11.2.1.7	Kageshima and Siraishi 法	373
11.2.1.8	電子系誘電率	373
11.2.1.9	非線形光学感受率	373
11.2.1.10	有効質量	376
11.2.2	格子系	377
11.2.2.1	概要	377
11.2.2.2	格子誘電率	377
11.2.2.3	ベリー一位相分極[16, 17, 18]	378
11.2.2.4	ボルン有効電荷	380
11.2.2.5	格子振動解析	380
11.2.2.6	圧電応答	381
11.2.2.7	圧電応答のベリー一位相理論[22]	381
11.2.2.8	圧電定数の計算方法	382
11.3	UVSOR-EPSILON	384
11.3.1	ファイルの設定	384
11.3.2	入力データの設定	384
11.3.2.1	Control ブロック	384
11.3.2.2	accuracy ブロック	384
11.3.2.3	epsilon ブロック	385
11.3.2.4	sw_epsilon	386
11.3.2.5	crystal_type	386
11.3.2.6	fermi_energy	386
11.3.2.7	photon	386
11.3.2.8	transition_moment	387
11.3.2.9	mass	387
11.3.2.10	BZ_integration	388
11.3.2.11	band_gap_correction	388
11.3.2.12	drude_term	388
11.3.2.13	nonlinear_optics	389
11.3.2.14	ipriepsilon	390
11.3.3	計算の実行	390
11.3.3.1	電荷密度の計算	390
11.3.3.2	誘電関数の計算	390
11.3.3.3	有効質量の計算	390
11.3.3.4	非線形光学感受率の計算	391
11.3.4	計算結果の解析	391
11.3.4.1	遷移モーメントの計算状況	391

11.3.4.2	電子状態.....	392
11.3.4.3	振動子強度の総和則[20]	392
11.3.4.4	誘電関数の計算結果.....	393
11.3.4.5	有効質量の計算結果.....	393
11.3.4.6	SHG 非線形光学感受率の計算結果.....	394
11.3.4.7	THG 非線形光学感受率の計算結果.....	395
11.3.5	計算例：Si2-電子誘電関数計算.....	397
11.3.5.1	電子密度の計算.....	397
11.3.5.2	誘電関数の計算.....	397
11.3.5.3	誘電関数の計算 1.....	399
11.3.5.4	バンド数の設定法.....	400
11.3.5.5	誘電関数の計算 2.....	400
11.4	UVSOR-BERRY-PHONON.....	402
11.4.1	入出力の説明.....	402
11.4.1.1	入力出力ファイル.....	402
11.4.1.2	入力ファイル"F_INP".....	402
11.4.1.3	入力および出力ファイル"F_BERRY".....	405
11.4.1.4	入力ファイル"F_EFFCHG".....	405
11.4.1.5	入力および出力ファイル"F_FORCE".....	405
11.4.1.6	出力ファイル"F_MODE".....	405
11.4.1.7	出力ファイル"F_EPSILON".....	406
11.4.1.8	ボルン有効電荷の出力.....	407
11.4.1.9	入力ファイル"F_STRFRC".....	408
11.4.2	計算例：水晶(α -quartz)の格子誘電率計算.....	408
11.4.2.1	計算手順.....	408
11.4.2.2	ベリー位相計算.....	408
11.4.2.3	ベリー位相計算の出力.....	414
11.4.2.4	格子振動解析.....	414
11.4.2.5	ボルン有効電荷の出力.....	417
11.4.2.6	格子振動解析と誘電率の出力.....	419
11.4.3	計算例：AlN の圧電定数の計算.....	420
11.4.3.1	計算手順.....	420
11.4.3.2	ベリー位相計算.....	420
11.4.3.3	格子振動解析.....	421
11.4.3.4	圧電定数のイオン固定項の計算.....	421
11.4.3.5	圧電定数の内部ひずみ項の計算.....	423
11.4.4	計算例：GaAs-格子誘電率計算.....	424
11.4.4.1	ベリー位相計算.....	424
11.4.4.2	振動解析および格子誘電率計算.....	425
11.5	UVSOR-BERRY-PHONON（改良版）.....	427
11.5.1	概要.....	427
11.5.2	使用方法.....	427
11.5.2.1	準備.....	427
11.5.2.2	コントロールファイルの記述.....	430
11.5.2.3	並列計算について.....	432
11.5.2.4	スクリプトの実行.....	432
11.5.2.5	計算のストップリスタート.....	433
11.5.2.6	注意点.....	433
11.5.3	計算例題：AlN の格子誘電率，圧電定数.....	434
11.5.3.1	AlN の格子誘電率.....	434

11.5.3.2	AIN の圧電定数のイオン固定項	436
11.5.3.3	AIN の圧電定数の内部ひずみ項	438
11.6	線形応答時間依存密度汎関数法 (LR-TDDFT)	441
11.6.1	機能の概要	441
11.6.1.1	はじめに	441
11.6.1.2	固体への適用	441
11.6.1.3	孤立系への適用	441
11.6.2	入力ファイルの記述	442
11.6.2.1	control ブロック	442
11.6.2.2	accuracy ブロック	443
11.6.2.3	structure ブロック	443
11.6.2.4	spectrum ブロック	443
11.6.3	計算の実行方法	445
11.6.4	出力ファイル	445
11.6.5	例題	445
11.6.5.1	Si 結晶の誘電スペクトル	446
11.6.5.2	C ₆ H ₆ 分子の光吸収断面積	446
11.6.6	使用上の注意点	447
11.7	付録	448
11.7.1	eps_file	448
11.7.2	nlo_file	448
11.8	参考文献	449

1. はじめに

1.1 PHASE システムの概要

PHASE システムは、電子状態計算プログラム（第一原理分子動力学法計算プログラム）PHASE/0、量子伝導特性計算プログラム ASCOT、原子全電子計算・擬ポテンシャル作成プログラム CIAO などのナノシミュレーションのプログラム・パッケージ群から構成されるシステムです。

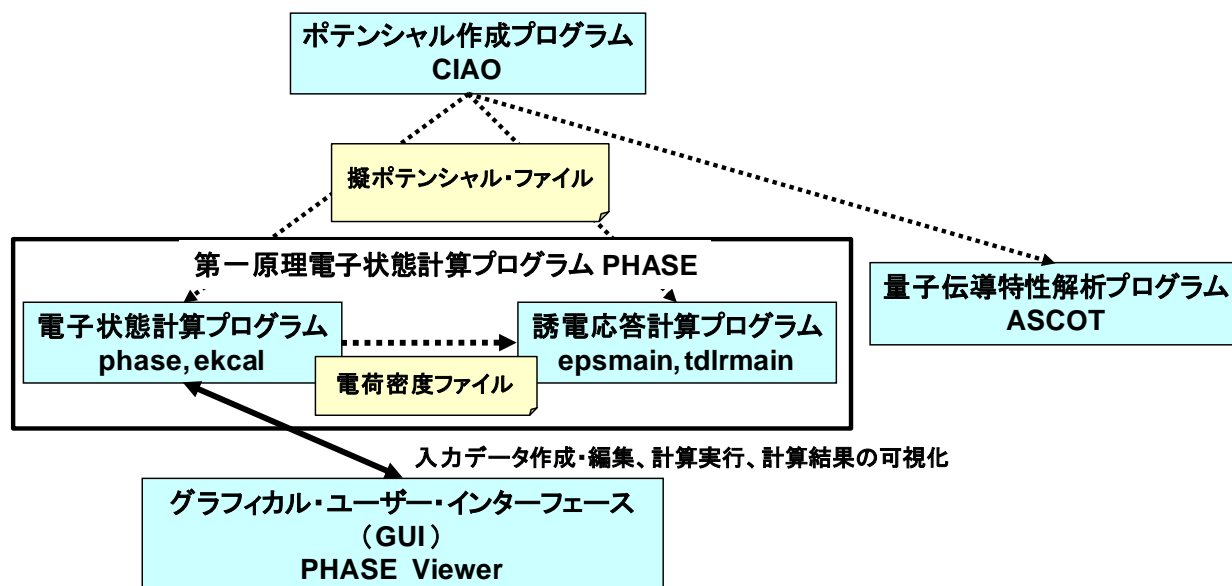


図 1-1 PHASE システムの主要なプログラム・パッケージ

表 1.1 PHASE システムの主要なプログラム・パッケージ

プログラム・パッケージ	プログラム	概要
PHASE/0 (第一原理電子状態計算プログラム)	phase, ekcal (旧 PHASE パッケージ)	密度汎関数理論に基づく擬ポテンシャル法による平面波基底の第一原理電子状態計算プログラムです。これらを使って、全エネルギー、電荷密度分布、電子の状態密度、バンド構造、安定な原子構造などの計算ができます。
	epsmain, tdlrmain (旧 UVSOR パッケージ)	Phase の計算結果を用いて、固体誘電体の誘電率を計算するプログラムです。これらのプログラムは、材料の電子系及び格子系の誘電率を計算できます。ゲート絶縁膜材料の誘電率をほぼ定量的に計算することができます。格子誘電率が大きい high-k 材料誘電率の予測及び解析に有効です。
CIAO (原子全電子計算・擬ポテンシャル作成プログラム)	ciao	CIAO は、密度汎関数理論に基づき、原子の全電子状態を第一原理計算し、得られた全電子ポテンシャルから擬ポテンシャルを計算するプログラムです。
ASCOT (量子伝導特性計算プログラム)	ascot	ASCOT は、非平衡グリーン関数法を用いて、電子状態および量子伝導特性を計算するプログラムです。
PHASE Viewer (GUI)	phase-viewer	PHASE の入力データ作成・編集、計算実行、計算結果の可視化を行うグラフィカル・ユーザー・インターフェース (GUI) です。

本マニュアルは、電子状態計算プログラム（第一原理分子動力学法計算プログラム）PHASE/0 と関連するツール群を対象としています。

1.2 PHASE とは？

1.2.1 PHASE の主な特徴、機能（PHASE で何が計算できるか？）

PHASE は、密度汎関数法、第一原理擬ポテンシャル法に基づく電子状態計算プログラムです。実験結果にフィッティングするパラメーターがないため、未知の物質に対する物性予測を（近似の範囲内で）精度良く計算することが出来ます。計算された電子状態（波動関数）を用いて様々な物理量の解析が可能です。原子配置に関しては、構造最適化、分子動力学法計算、化学反応経路解析など様々な解析が可能です。PHASE の主な機能を以下の表にまとめます。

主な機能	得られる物理量など (現象・挙動、物性など)
電子状態計算	状態密度分布 (DOS) バンド構造 電荷密度分布
エネルギー、力計算	エネルギー、原子に働く力 格子定数、弾性パラメーター 応力テンソル
構造最適化 (緩和) 分子動力学法計算	安定な原子配置 原子の運動
振動解析	振動数・振動モード
陽電子寿命解析	陽電子寿命
STM 像解析	STM 像
化学反応解析	化学反応経路、エネルギー障壁

PHASE の主な特徴を以下にまとめます。

計算手法		
	第一原理計算	実験によるパラメータフィットがないため、未知の物質や実験の困難な系に対する物性予測が可能 ハイブリッド汎関数などの高精度な電子状態計算手法を利用することも可能
	密度汎関数法	物性分野での解析に広く使用されている LDA 法や GGA 法などの密度汎関数法による信頼性の高い計算が可能
	擬ポテンシャル	イオンコアの影響を擬ポテンシャルによって取り扱うことにより、高精度な解析計算が可能
計算機能		
	物性解析機能	多様な物性解析機能を利用することにより、様々な実験との比較が可能
	構造解析機能	構造最適化、分子動力学法による時間発展、反応経路解析などの原子配置に関する解析が可能
	大規模計算	MPI, OpenMP を利用した並列計算により、数十万コア以上の超大規模並列計算まで対応
ユーザーフレンドリ		
	入力データ	入力ファイルをタグ形式にして、初めて使用するユーザーでも、入力パラメータの物理的意味が理解し易いように構成。 様々な設定が可能であると同時に、ほとんどのパラメータに既定値が設定されているため、非常にシンプルな設定でも計算することが可能。
	ツール	バンド構造、状態密度、電荷密度分布などの計算結果をグラフ化するツールなど、結果の解析に便利なツールが付属
	動作環境	個人ユースのパソコン (WindowsPC) から最先端のスパコンまで
	ユーザインターフェース	GUI (PHASE-Viewer) による操作が可能

1.2.2 PHASE/0 のプログラム構成

プログラムパッケージ PHASE/0 には、以下のプログラム、ツールなどが含まれています。

プログラムパッケージ PHASE/0		概要
プログラム	phase	SCF 計算、分子動力学法計算を行います。また収束した電荷密度分布から状態密度やバンド分散を計算することができます。
	ekcal	状態密度計算、バンド計算において k 点の個数が多い場合に使う補助プログラムとして ekcal があります。これらの処理を簡便に行うための補助スクリプトファイルがいくつか用意されています。
	epsmain	Phase の計算結果に基づき、電子系誘電関数を計算します。
	tdlrmmain	Phase の計算結果に基づき、線形応答時間依存密度汎関数法 (LR-TDDFT) により誘電関数を計算します。
ツール	band_kpoint.pl	バンド計算用の k 点のリストを生成する Perl スクリプト
	dos.pl	状態密度のグラフ (EPS 画像) を作成する Perl スクリプト
	band.pl	バンド構造のグラフ (EPS 画像) を作成する Perl スクリプト

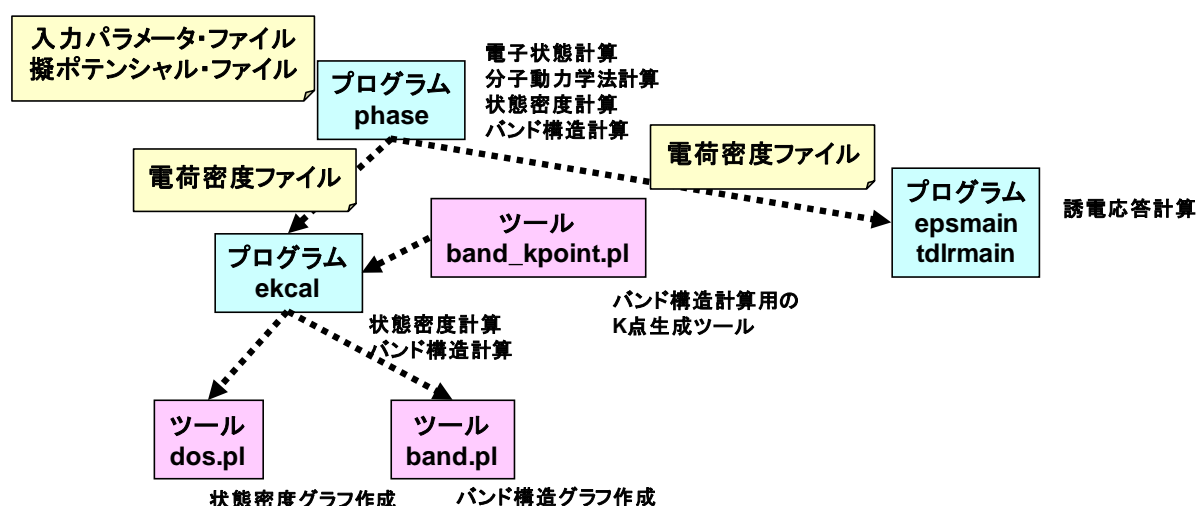


図 1-2 PHASE/0 のプログラム構成

1.2.3 利用可能な動作環境（計算機環境要件）

PHASE/0 プログラムは fortran90 と C で記述されています。これらのコンパイラが使える計算機システムが必要です。大学の計算機センターなどの一般利用可能なシステムでは通常使用することが可能です。並列計算をする場合には MPI ライブラリがインストールされている必要があります。

必要（利用可能）なソフトウェア、ライブラリ

- Fortran90 コンパイラ、C コンパイラ（必須）
- MPI ライブラリ（並列計算に必須）
- 行列演算ライブラリ LAPACK, BLAS（オプション）
- FFT ライブラリ FFTW（オプション）
- Perl（オプション）・・・PHASE ツールで必要
- Gnuplot（オプション）・・・PHASE ツールで必要

PHASE/0 が手軽に利用出来るように、Windows 用の実行ファイルも配布しています。但し、非並列版のため、規模の大きな計算には難があります（実行出来ない、時間が膨大にかかる、など）。

本マニュアルで記述する操作についてはLinux (Unix) を使用しているものとしています。お使いのシステムが異なる場合には、適宜読み替えてください。

1.2.4 PHASE/0 の機能 (2D 並列版と 3D 並列版の比較)

PHASE には、2D 並列版と 3D 並列版の 2 つの並列版プログラムがあります。

	並列化手法	ソースプログラム
2D 並列版	k 点並列 + エネルギー (バンド) 並列	src_phase
3D 並列版	k 点並列 + エネルギー (バンド) 並列 + G 点並列	src_phase_3d

機能	2D 並列版	3D 並列版
構造最適化	✓	✓
単位胞最適化	✓	✓
状態密度	✓	✓
局所状態密度	✓	✓
射影状態密度	✓	✓
電荷密度出力	✓	✓
部分電荷密度出力	✓	✓
最大局在化ワニア関数	✓	
バンド構造	✓	✓
実空間法	✓	✓
ストレステンソル	✓	✓
仕事関数	✓	✓
XPS 解析	✓	✓
振動解析, フォノンバンド	✓	✓
分子動力学	✓	✓
DFT+U 法	✓	✓
ハイブリッド汎関数法	✓	✓
ESM 法	✓	✓
DFT-D2 法	✓	✓
vdW-DF 法	✓	✓
電荷密度予測	✓	✓
波動関数予測	✓	✓
NEB 法	✓	✓
blue moon 法	✓	✓
メタダイナミクス法	✓	✓
RTP-TDDFT	✓	✓
LR-TDDFT	✓	
ノンコリニア磁性	✓	
スピン軌道相互作用	✓	
陽電子寿命解析	✓	
PAW 法	✓	✓
電子系誘電関数	✓	✓
ボルン有効電荷	✓	✓
格子系誘電関数	✓	✓
圧電定数	✓	✓

1.3 マニュアルの構成

マニュアルは、以下のような章構成になっています。

1 章	はじめに	PHASE システム、プログラムパッケージ PHASE/0 の概要について説明しています。
2 章	PHASE/0 の基本的な利用方法	PHASE/0 の計算手順などの最も基本的な利用方法を説明しています。PHASE/0 の計算の流れが概観できます。
3 章	入力パラメータファイル：NFINP.DATA（詳細版）	入力パラメータファイルのリファレンスマニュアルとして使用できます。多くのパラメータについては知らなくても PHASE/0 は利用可能です。高度な利用をする際などに参照すると良いと思われます。
4 章	基本機能を利用した計算例	PHASE/0 の基本的な機能を利用した計算例を幾つか示しています。チュートリアルとして使用できます。ここを読みながら 3 章の必要な項目を参照すると良いかも知れません。
5 章	応用機能	応用機能を説明しています。各機能を利用する際に有用です。
6 章	PAW 法による計算	PHASE/0 では PAW 法を使用することが出来ます。PAW 法の利用方法、適用範囲について説明します。
7 章	計算精度など補足事項	上記の章を補足する様々な項目があります。必要に応じて参照ください。
8 章	PHASE/0 のインストール	バイナリプログラムを作る（コンパイルする）方法を説明しています。
9 章	PHASE/0 プログラム、ツールの実行方法	PHASE/0 のバイナリプログラムやツールの使い方を説明します。
10 章	入出力ファイル	PHASE/0 の入出力ファイルについて説明します。
11 章	誘電応答解析機能 UVSOR	誘電応答解析を行う方法を説明します。

初めて本マニュアルを読む方は、2 章に続けて 4 章を読むことを推奨します。4 章を読む際に出てきた入力パラメータについては 3 章を参照してください。その後、5 章以降については必要に応じて読むことを推奨します。

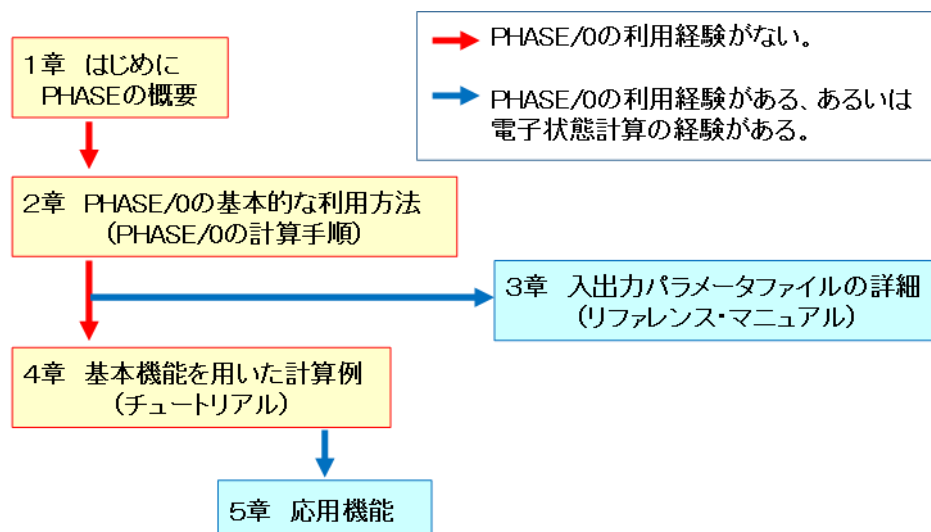


図 1 3 マニュアルを読む順番

1.4 PHASE の更新履歴

バージョン 8.00 2009/06 公開	<ul style="list-style-type: none"> ・拘束条件付きの動力学を追跡する機能が追加されました。 ・DFT+U 法による構造最適化/分子動力学シミュレーションに対応しました。
バージョン 8.01 2010/03 公開	<ul style="list-style-type: none"> ・BLAS を利用した高速化に対応しました。
バージョン 9.00 2010/06 公開	<ul style="list-style-type: none"> ・キャッシュチューニング、BLAS 化をさらに進め、高速化を行いました。 ・ファンデルワールス相互作用を考慮することができるようになりました。 ・自由エネルギー計算が行えるようになりました。 ・DFT+U 法を利用して、バンド構造の計算ができるようになりました。 ・ハイブリッド汎関数が利用できるようになりました。
バージョン 10.00 2011/06 公開	<ul style="list-style-type: none"> ・電子状態計算の収束性が向上しました。 ・PAW 法に対応しました。 ・メタダイナミクスに対応しました。 ・van der Waals 密度汎関数のセルフコンシステントな実装を行いました。 ・構造最適化に BFGS 法が利用できるようになりました。 ・PHASE TOOLS に新たなスクリプトを加えました。 ・擬ポテンシャルの読み込みに関する不具合を修正しました。 <p><u>この修正によって、従来のバージョンと全エネルギーの絶対値は一致しなくなります。</u></p>
バージョン 10.01 2011/08 公開	<ul style="list-style-type: none"> ・スピンを考慮している系の収束性が向上しました。 ・GGA に関する不具合を修正しました。 <p><u>この修正によって、従来のバージョンと全エネルギーの絶対値は一致しなくなります。</u></p>
バージョン 11.00 2012/06 公開	<ul style="list-style-type: none"> ・新しい波動関数ソルバーを導入しました。 ・ハイブリッド汎関数の計算機能を改良しました。 ウルトラソフト擬ポテンシャル対応、k 点を間引いた計算対応など ・構造最適化機能の GDIIS、BFGS 法の継続計算に対応しました。 ・ウルトラソフト擬ポテンシャルを用いた局所状態密度計算を高速化しました。 ・構造最適化、分子動力学法計算の計算中における状態密度、電荷密度の出力に対応しました。 ・多くの不具合を修正しました。 ・3 軸並列 (G 点並列) 版を公開しました。

PHASE/O 2014.01 2014/04 公開	<ul style="list-style-type: none"> ・波動関数ソルバー、電荷密度混合法の自動設定機能を追加しました。 ・構造更新時の波動関数、電荷密度の予測機能を追加しました。 ・時間依存型密度汎関数法計算を追加しました。 ・有効遮蔽体 (ESM) 法等の計算機能のインターフェースを追加しました。 ・ユニットセル最適化機能等を追加しました。 ・仕事関数の計算機能を追加しました。 ・波動関数ソルバー (修正 Davidson 法等) を改良 (高速化) しました。 ・ハイブリッド汎関数の計算機能を改良 (高速化) しました。 ・ファンデルワールス相互作用の計算機能を改良 (高速化) しました。 ・ノンコリニア系の計算、スピン軌道相互作用計算機能を追加しました。 ・原子配置のチェック機能を追加しました。 ・構造最適化機能に CG 法の改良版を追加しました。 ・フォノン計算機能を改良しました。 ・非局所ポテンシャルや欠損電荷の演算を実空間で行うことができるようにしました。 ・誘電応答解析プログラム UVSOR を統合しました。 ・3 軸並列 (G 点並列) 版の機能を拡充しました。 ・不具合を修正しました。
PHASE/O 2014.02 2014/07 公開	<ul style="list-style-type: none"> ・ESM 法を 3 軸並列 (G 点並列) 版に実装しました。 ・ハイブリッド汎関数の使用方法を改善しました。 ・ハイブリッド汎関数の計算機能を改良しました。ハイブリッド汎関数の計算において、波動関数ソルバー pdavidson 法と pkosugi を使用可能にしました。 ・不具合を修正しました。 ・マニュアルを改訂しました。
PHASE/O 2014.03 2014/12 公開	<ul style="list-style-type: none"> ・ハイブリッド汎関数計算機能を高速化・省メモリ化しました。(2 軸並列版)。 ・vdW-DF 計算機能を高速化しました。(2 軸並列版)。 ・非局所ポテンシャルの実空間積分機能を 3 軸並列 (G 点並列) 版に実装しました。 ・波動関数予測機能を 3 軸並列 (G 点並列) 版に実装しました。
PHASE/O 2015.01 2015/10 公開	<ul style="list-style-type: none"> ・ELNES / XANES 解析機能を実装しました。 ・バンド構造を群論に基づいて簡約化するプログラム band_symm を追加しました。 ・計算事例を追加しました。(スピン軌道相互作用をとりいれたバンド構造計算および XANES 解析計算) ・PAW 法の不具合を修正しました。この修正によって、系によっては以前よりも収束性が向上しました (sw_PAW=on で DFT+U あるいは GGA+U 法を使っている場合、以前の版ではこの版よりも全エネルギーが高くなる場合があります)。
PHASE/O 2016.01 2017/05 公開	<ul style="list-style-type: none"> ・化学ポテンシャル一定のシミュレーション法を実装しました。 ・ハイブリッド汎関数計算機能を高速化しました。(3 軸並列版) ・不具合を修正しました。
PHASE/O 2017.01 2018/02 公開	<ul style="list-style-type: none"> ・圧力一定の分子動力学シミュレーション法を実装しました。 ・3 軸並列版にストレステンソル, vdW-DF 計算機能を加えました。 ・ボルン電荷を簡単に計算できるようにしました。 ・双極子補正機能に対応しました。 ・バンド対称化プログラム(band_symm)を更新しました。 ・不具合を修正しました。
PHASE/O 2018.01 2018/08 公開	<ul style="list-style-type: none"> ・以下の機能が三次元版で利用できるようになりました。 <ul style="list-style-type: none"> ➢ ベリー位相計算機能 ➢ PDOS ➢ 固定電荷計算 ・3D 版に、低並列で利用する場合の高速化のオプションを追加しました。 ・Atomic Simulation Environment (ASE)用の Python スクリプトを追加しました。 ・SC-DFT 法に対応しました。 ・ハイブリッド汎関数法を高速化しました。

	<ul style="list-style-type: none"> • PAW 法に非球面積分のオプションを追加しました。 • ESM 法を高速化しました。 • 不具合を修正しました。
PHASE/0 2019.01 2019/04 公開	<ul style="list-style-type: none"> • 分子動力学シミュレーション機能を改良しました。 <ul style="list-style-type: none"> ➤ Nose-Hoover chain 法が使えるようになりました。 ➤ “温度プロファイル”を設定することができるようになりました。 ➤ 原子群をある領域に閉じ込めることができるようになりました。 • BoltzTraP に対応しました。 • DFT-D3 法に対応しました。 • ストレステンソル補正機能を改良しました。 • ASL の FFTW ラッパーが使えるようになりました。

2. PHASE/0 の基本的な利用方法

ここでは、PHASE/0 の基本的な利用方法を説明します。計算の手順を優先的に説明するために、詳細な説明は省いています。詳細な説明が必要な部分については3章以降を参照してください。

この章以降については PHASE/0 をインストールしていることを前提とします。PHASE/0 のインストールについては第8章を参考にしてください。

2.1 PHASE/0 の計算手順の概要

PHASE/0 に含まれるプログラム phase で計算を実行するための大まかな手順は以下のようになっています。

- ① 入力データの準備
- ② 計算の実行
- ③ 計算結果の確認
- ④ 計算結果の解析、可視化

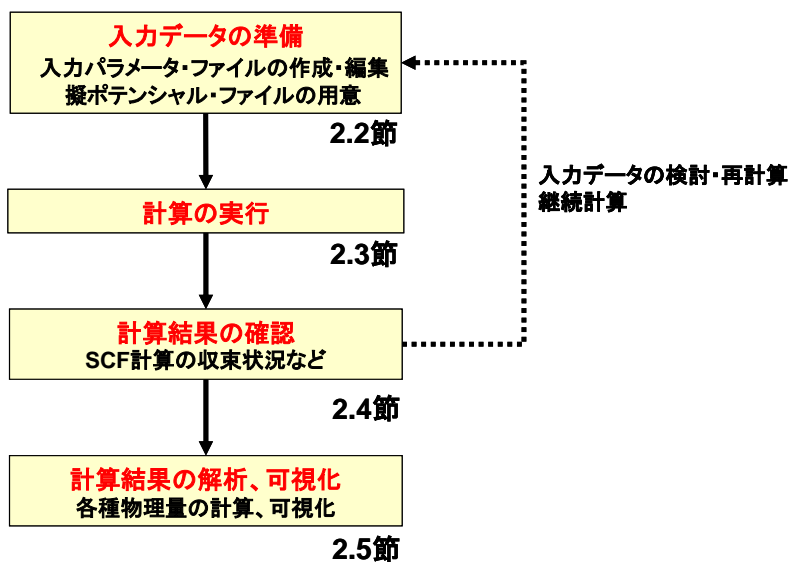


図 2-1 PHASE を用いた電子状態計算の手順の概要

プログラム ekcal による計算は、phase で SCF 計算を収束させ電荷密度分布ファイルを出力したのちに行います。

2.2 入力データの準備

2.2.1 計算に最低限必要なファイル

PHASE 計算を実行するために最低限必要なファイルは、入力パラメータ・ファイルと擬ポテンシャル・ファイルの2つです。これらのファイルを用意して実行ディレクトリに置いてください。

ファイル名を変更したい場合やファイルを実行ディレクトリ以外に置きたい場合は、入出力ファイル設定ファイル `file_names.data` も用意してください。

入力データ

ファイル	概要
入力パラメータファイル	<p>「どのようなモデル（原子配置など）に対して、どのような計算を行うか」を指定するファイルです。</p> <p>デフォルトのファイル名は“<code>nfinp.data</code>”です。</p> <p>既定値を多用したシンプルな入力については第 2.2.2 節で解説します。高度な利用が可能となる詳細な解説については第 3 章を参照ください。また、具体的な利用例については第 4 章、応用機能については第 5 章を参照ください。</p>
擬ポテンシャルファイル	<p>各元素の擬ポテンシャルのファイルです。詳細については 2.2.3 節で解説します。</p> <p>利用する元素に応じてあらかじめ用意（ダウンロードまたは作成）しておく必要があります。デフォルトのファイル名は <code>pot.01</code>, <code>pot.02</code>...です。ダウンロードしたファイルを使用する際にはファイル名を変更してください。デフォルトファイル名を使用しない場合には <code>file_names.data</code> ファイルでファイル名を設定してください。</p> <p>最大で 16 種まで設定可能です。</p>

入出力ファイル設定ファイル

<code>file_names.data</code>	<p>ファイル名の指定に利用するファイルです。無くても PHASE を使用することは可能です。その場合は全てのファイル名が既定値となります。</p> <p>このファイルを使用することにより、i)ファイル名、ii)ファイルの置き場所をユーザーが自由に設定することが可能となります。</p> <p>このファイルを使用する場合は、必ずこのファイル名で（変更不可）、実行ディレクトリに置く必要があります。詳細については 2.2.4 節で解説します。</p>
------------------------------	--

2.2.2 入力パラメータファイル : nfinp.data (簡略版)

入力パラメータファイル (nfinp.data) は、計算系 (原子構造) や計算手法・計算条件の計算パラメータの設定を行います。入力パラメータファイルの詳細については、3 章を参照ください。多くのパラメータにはデフォルト値が設定されていますので、それらを利用することによりユーザーが最低限指定しなければならない部分は非常に少なくできます。ここではほぼ最小限の入力パラメータファイルについて説明します。

2.2.2.1 入力パラメータファイル例

入力パラメータファイルは、タグ (キーワード) と中括弧 {} で囲まれたブロックの階層構造となっています。計算パラメータについては、ブロック内でキーワードと値で指定します。

Si ダイヤモンド結晶 (2 原子) の電子状態計算を行う場合の基本的な計算条件を記述した入力ファイル例です。

```
control{
  condition = initial
  cpumax = 86400 sec
  max_iteration = 10000
}

accuracy{
  cutoff_wf = 25.0 rydberg
  cutoff_cd = 100.0 rydberg
  num_bands = 8
  ksampling{
    method = monk
    mesh{
      nx = 10
      ny = 10
      nz = 10
    }
  }
  initial_wavefunctions = atomic_orbitals
  initial_charge_density = atomic_charge_density
  scf_convergence{
    delta_total_energy = 1e-10
    succession = 3
  }
  force_convergence{
    max_force = 0.001 hartree/bohr
  }
}

structure{
  element_list{
    #tag    element    atomicnumber
        Si      14
  }
  unit_cell{
    #units angstrom
    a_vector = 0 2.732299538 2.732299538
    b_vector = 2.732299538 0 2.732299538
    c_vector = 2.732299538 2.732299538 0
  }
  unit_cell_type = bravais
  atom_list{
    atoms{
```



```

    #tag    element    rx    ry    rz    mobile
        Si    0.125 0.125 0.125    0
        Si   -0.125 -0.125 -0.125    0
    }
    coordinate_system = internal
}

wavefunction_solver{
    solvers{
        #tag    sol    till_n    prec    cmix    submat
        pdavidson    1    on    1    on
        rmm3          -1    on    1    on
    }
    rmm{
        edelta_change_to_rmm=5e-5
    }
}

charge_mixing{
    mixing_methods{
        #tag    n0    method    rmxs    rmxe    istr    prec    nbmix
        1    pulay    0.40    0.40    3    on    15
    }
}

Postprocessing{
    dos{
        sw_dos = ON
        deltaE = 1.e-4 hartree
    }
    charge{
        sw_charge_rspace    = ON
        filetype = cube    !{cube|density_only}
        title = "This is a title line for the bulk Si"
    }
}
}

```

最上位のブロックは、以下のものがあります。

control ブロック	全体的な計算条件の設定
accuracy ブロック	計算精度の設定
structure ブロック	原子構造の設定
wavefunction_solver ブロック	波動関数ソルバーの設定
charge_mixing ブロック	電荷密度混合法の設定
structure_evolution ブロック	構造最適化、分子動力学法計算の設定
postproccesing ブロック	後処理の設定
printlevel ブロック	ログ出力の設定

以下の節で、各ブロックの入力パラメータを簡単に説明します。

2.2.2.2 Control ブロック

control ブロックでは、計算全体の制御に関するパラメータを指定します。

```
control{
  condition = initial
  cpumax = 86400 sec
  max_iteration = 10000
}
```

condition 最初の計算、継続（リスタート）計算、電荷を固定した計算などの計算モードを指定します。
initial は最初の計算、**continuation** は継続計算になります。

cpumax 計算時間（計算を打ち切る時間）を指定します。

max_iteration SCF 計算を打ち切るイタレーション数を指定します。

2.2.2.3 Accuracy ブロック

accuracy ブロックでは、計算精度に関するパラメータを指定します。

```
accuracy{
  cutoff_wf = 25.0 rydberg
  cutoff_cd = 100.0 rydberg
  num_bands = 8
  ksampling{
    method = monk
    mesh{
      nx = 10
      ny = 10
      nz = 10
    }
  }
  initial_wavefunctions = atomic_orbitals
  initial_charge_density = atomic_charge_density
  scf_convergence{
    delta_total_energy = 1e-10
    succession = 3
  }
  force_convergence{
    max_force = 0.001 hartree/bohr
  }
}
```

cutoff_wf 波動関数のカットオフエネルギーを指定します。

cutoff_cd 電荷密度のカットオフエネルギーを指定します。

num_bands 計算するバンド数を指定します。

ksampling ブロック **k** 点サンプリングを指定します。

method **k** 点サンプリングの方法を指定します。**monk** は Monkhorst-Pack 法[\[1\]](#)によるサンプリングです。

mesh 逆空間の分割数を指定します。

initial_wavefunctions 波動関数の初期値の計算方法を指定します。
atomic_charge_density は、擬ポテンシャルファイルの原子の電荷密度から初期値を計算します。

scf_convergence ブロック	エネルギーによる SCF 計算の収束判定条件を指定します。
delta_total_energy	エネルギーの変化量の閾値を指定します。エネルギーの変化量が、この閾値より小さい場合、SCF 計算は収束したと判定します。
force_convergence ブロック	原子に働く力による SCF 計算の収束判定条件を指定します。
max_force	原子に働く力の最大値の閾値を指定します。原子に働く力の最大値が、この閾値より小さい場合、SCF 計算は収束したと判定します。

2.2.2.4 Structure ブロック

structure ブロックでは、原子構造を指定します。

```

structure{
  element_list{
    #tag    element    atomicnumber
        Si     14
  }
  unit_cell{
    #units angstrom
    a_vector = 0 2.732299538 2.732299538
    b_vector = 2.732299538 0 2.732299538
    c_vector = 2.732299538 2.732299538 0
  }
  unit_cell_type = bravais
  atom_list{
    atoms{
      #tag    element    rx    ry    rz    mobile
          Si     0.125  0.125  0.125    0
          Si    -0.125 -0.125 -0.125    0
    }
    coordinate_system = internal
  }
}

```

element_list ブロック	計算する系の元素のリストを指定します。 この例では、元素は Si (シリコン)、その原子番号が”14”であることを意味しています。
unit_cell ブロック	ユニットセルのサイズ、形状を指定します。 #units angstrom は、単位がオングストロームを意味します。 a_vector、b_vector、c_vector は、格子ベクトルを指定します。
atom_list ブロック	元素と座標を指定します。 この例では、Si 原子が 2 個あり、その内部座標が 0.125, 0.125, 0.125 としています。
coordinate_system	原子座標系を指定します。 internal は、ユニットセルを基準とした内部座標系であることを意味しています。

2.2.2.5 Wavefunction_solver ブロック

wavefunction_solver ブロックでは、波動関数の更新方法に関するパラメータを指定します。

```

wavefunction_solver{
  solvers{

```

```

        #tag    sol    till_n  prec  cmix  submat
        pdavidson 1    on  1    on
        rmm3      -1    on  1    on
    }
    rmm{
        edelta_change_to_rmm=5e-5
    }
}

```

solvers ブロック 波動関数ソルバーを指定します。
この例では、最初の波動関数ソルバーは **pdavidson** 法、2 番目の波動関数ソルバーは **RMM** 法としています。

rmm ブロック

edelta_change_to_rmm 波動関数ソルバーを **RMM** 法に変更する閾値を指定します。ここで指定する値よりもエネルギーの変化量が小さくなると **RMM** 法に移行します。

2.2.2.6 Charge_mixing ブロック

charge_mixing ブロックでは、電荷密度の更新方法（混合方法）に関するパラメータを指定します。

```

charge_mixing{
    mixing_methods{
        #tag no  method    rmxs  rmxe  istr  prec  nbmix
        1  pulay  0.40  0.40  3     on   15
    }
}

```

mixing_methods ブロック 電荷密度の更新方法（混合方法）を指定します。この例では **pulay** 法[2]を指定しています。この他に **broyden2** 法[3]、**simple mixing** 法を指定することができます。

2.2.2.7 Postprocessing ブロック

Postprocessing ブロックでは、後処理に関するパラメータを指定します。

```

Postprocessing{
    dos{
        sw_dos = ON
        deltaE = 1.e-4 hartree
    }
    charge{
        sw_charge_rspace = ON
        filetype = cube
        title = "This is a title line for the bulk Si"
    }
}

```

dos ブロック 状態密度の計算パラメータを指定します。
sw_dos **ON** で状態密度の計算を行います。
deltaE 状態密度のエネルギー間隔を指定します。

charge ブロック 電荷密度分布の出力パラメータを指定します。

sw_charge_rspace	ON で出力用の電荷密度分布の計算を行います。
filetype	電荷密度分布のファイル形式を指定します。cube は Gaussian CUBE 系形式を指定しています。
title	出力の Gaussian CUBE 系形式のファイルにおけるタイトルを指定します。

2.2.2.8 入力パラメータファイルの最低限の設定項目

ここで紹介した例では設定項目がそれなりにありますが、大部分のパラメータは多くの計算で共通に使用できます。そのため、異なる系を計算する場合に、ユーザーがわざわざ変更する必要はありません。

ユーザーが必ず設定しなければならないのは、カットオフエネルギー、 k 点、原子構造に関する設定です。入力パラメータファイルの Accuracy ブロックにあるカットオフエネルギーの指定 cutoff_wf, cutoff_cd, k 点の指定 ksampling, Structure ブロックにある元素の指定 element_list, ユニットセルの指定 unit_cell, 原子座標の指定 atom_list の部分です。

先程の入力パラメータファイルの例において、ユーザーの設定部分を網掛けで示します。ユーザが計算したい系にあわせて、この部分を変更することにより、様々な系の計算をすることができます。ただし、最適な計算条件については、個々のパラメータを適切に設定する必要があります。

```
control{
  condition = initial
  cpumax = 86400 sec
  max_iteration = 10000
}

accuracy{
  cutoff_wf = 25.0 rydberg
  cutoff_cd = 100.0 rydberg
  num_bands = 8
  ksampling{
    method = monk
    mesh{
      nx = 10
      ny = 10
      nz = 10
    }
  }
  initial_wavefunctions = atomic_orbitals
  initial_charge_density = atomic_charge_density
  scf_convergence{
    delta_total_energy = 1e-10
    succession = 3
  }
  force_convergence{
    max_force = 0.001 hartree/bohr
  }
}

structure{
  element_list{
    #tag      element      atomicnumber
      Si      14
  }
  unit_cell{
    #units angstrom
```

```

a_vector = 0 2.732299538 2.732299538
b_vector = 2.732299538 0 2.732299538
c_vector = 2.732299538 2.732299538 0
}
unit_cell_type = bravais
atom_list{
  atoms{
    #tag    element    rx    ry    rz    mobile
    Si      0.125    0.125    0.125    0
    Si     -0.125   -0.125   -0.125    0
  }
  coordinate_system = internal
}
}

wavefunction_solver{
  solvers{
    #tag    sol    till_n    prec    cmix    submat
    davidson    1    on    1    on
    rmm3        -1    on    1    on
  }
  rmm{
    edelta_change_to_rmm=5e-5
  }
}

charge_mixing{
  mixing_methods{
    #tag    no    method    rmxs    rmxe    istr    prec    nbmix
    1    pulay    0.40    0.40    3    on    15
  }
}

Postprocessing{
  dos{
    sw_dos = ON
    deltaE = 1.e-4 hartree
  }
  charge{
    sw_charge_rspace = ON
    filetype = cube !{cube|density_only}
    title = "This is a title line for the bulk Si"
  }
}
}

```

ちなみに、上記の入力パラメータファイルに対し、最低限の設定項目のみを記述し、他の計算条件を省略（デフォルトを使用）した以下のような入力パラメータファイルでも、計算は可能です。上記の入力パラメータでは計算条件がデフォルトとは異なる部分があるため、上記と下記の入力パラメータファイルを用いた計算では、計算途中の状況が異なります。

```

accuracy{
  cutoff_wf = 25.0 rydberg
  cutoff_cd = 100.0 rydberg
  ksampling{
    mesh{
      nx = 10
      ny = 10
      nz = 10
    }
  }
}

```

```

}
}
structure{
  element_list{
    #tag      element      atomicnumber
        Si      14
  }
  unit_cell{
    #units angstrom
    a_vector = 0 2.732299538 2.732299538
    b_vector = 2.732299538 0 2.732299538
    c_vector = 2.732299538 2.732299538 0
  }
  atom_list{
    atoms{
      #tag      element      rx      ry      rz      mobile
        Si      0.125  0.125  0.125      0
        Si     -0.125 -0.125 -0.125      0
    }
  }
}

```

2.2.3 擬ポテンシャルファイル

擬ポテンシャルファイルは、元素ごとに用意する必要があります。例えば、水 (H_2O) の計算をする場合には、H 原子と O 原子の擬ポテンシャルファイルが必要になります。擬ポテンシャルファイルはポータルサイト (<https://azuma.nims.go.jp/>) からダウンロードできます。自分でポテンシャルを作成する場合には、PHASE の関連プログラム CIAO で作成できます。CIAO については CIAO のマニュアルを参照してください。

2.2.3.1 擬ポテンシャルの種類（内殻の取り扱い方法）：フローズンコア型と PAW 型

PHASE の擬ポテンシャルは大きく分けてフローズンコア型と PAW 型[4]の2種類があります。

フローズンコア（凍結核）型	内殻電子と核をイオン芯として完全に凍結させた状態として扱い、価電子状態だけを取り扱います。これはさらにノルム保存擬ポテンシャル[5]とウルトラソフト擬ポテンシャル[6]に分類できます。
PAW 型	PAW (Projector Augmented Wave) 型のポテンシャル[4,7]では、内殻の電子密度を考慮した計算をします。

フローズンコア（凍結核）型と PAW ポテンシャルを混在させて計算することはできませんので注意してください。

2.2.3.2 擬ポテンシャルファイルの入手方法

ポータルサイト (<https://azuma.nims.go.jp/>) から、周期表にある全ての元素の擬ポテンシャルファイルをダウンロードすることが可能です。

公開している擬ポテンシャルファイルは、以下の命名規則にしたがってファイル名が決められています。

元素名_交換相関エネルギーの計算方法_（PAW 型）擬ポテンシャル型_識別用の数字.pp
--

例えば Si_ggapbe_paw_nc_01.pp という擬ポテンシャルファイルは、シリコンに対応する、一般化密度勾配近似(GGA)法[8]によって交換相関エネルギーを計算する、PAW 型で、ノルム保存擬ポテンシャル[5] (nc は norm conserving (ノルム保存) の略) の、01 番（一般に作成された順番で番号が付けられます）ということを表します。交換相関エネルギーの計算方法には一般化密度勾配近似法のほかに局所密度近似(LDA)法があり、ggapbe の部分が ldapw91[9]となります。擬ポテンシャル型がウルトラソフト型[6]の場合には nc の部分が us (ultrasoft の略) となります。

元素名：	シリコン
交換相関エネルギーの計算方法	ggapbe 法
擬ポテンシャル型：	PAW 型でノルム保存擬ポテンシャル (nc は norm conserving (ノルム保存) の略)
識別用の番号：	データの通し番号

2.2.3.3 擬ポテンシャルの指定方法

デフォルトの擬ポテンシャルファイル名は、入力パラメータファイルのタグ element_list で設定した元素順に pot.01, pot.02....です。

file_name.data ファイルを用いた場合は、擬ポテンシャルのディレクトリ、ファイル名を指定できます。

2.2.4 入出力ファイル設定ファイル file_names.data

入出力ファイル設定ファイル **file_names.data** は入力パラメータファイルなどのファイル名を設定するために使用します。無くても **PHASE** を使用することは可能です。その場合は全てのファイルがデフォルトのファイル名になります。

このファイルを使用することにより、i)ファイル名、ii)ファイルの置き場所、をユーザーが自由に設定することが可能になります。「**file_names.data**」そのもののファイル名は、変更不可です。

このファイルを使用する場合は、必ずこのファイル名で実行ディレクトリに置く必要があります。

file_names.data のフォーマットは、以下のようになります。

```
&fnames
ファイル名キーワード = 'ファイル名 (ファイルへのパス) '
...
...
/
```

最後に、“/” が必要な点に注意してください。例えば、以下のように記述します。

```
&fnames
F_INP = './nfinp.data'
F_POT(1) = './Si_ggapbe_nc_01.pp'
F_POT(2) = './O_ggapbe_us_02.pp'
F_CHGT = './nfchgt.data'
F_CHR = './nfchr.cube'
/
```

ファイルへのパスは、実行ディレクトリからの相対パスでも、絶対パスでも使用できます。

F_POT(n)は、入力パラメータファイルにおいて **n** 番目に指定された原子種に対する擬ポテンシャルファイルを指定します。この例では、1 番目に定義された元素 **Si** の擬ポテンシャルファイルが **Si_ggapbe_nc_01.pp**、2 番目に定義された元素 **O** の擬ポテンシャルファイルが **O_ggapbe_us_02.pp** というファイルになります。

利用できるファイル名キーワードやデフォルトのファイル名を表 2.1 に示します。

表 2.1 入出力ファイル設定ファイル **file_names.data** で設定可能なファイル一覧

ファイル名 キーワード	対応する プログラム	入出力 の区別	デフォルトの ファイル名	概要
F_INP	phase ekcal	入力	nfinp.data	入力パラメータ・ファイル
F_POT(n)	phase ekcal	入力	pot.01, pot.02, ...	擬ポテンシャル・ファイル。この配列に与えられた名前のファイルを読みこみます。各元素に1つのポテンシャルファイルが必要になります。
F_STOP	phase ekcal	入力	nfstop.data	途中で計算を停止させる回数を記述するファイル。
F_KPOINT	phase ekcal	入力	kpoint.data	サンプリング k 点を設定するファイル。
F_DYNM	phase	出力	nfdynm.data	構造緩和計算、分子動力学計算などにおいて、原子の座標と力を出力するファイル。

F_ENF	phase	出力	nfefn.data	全エネルギーと力の最大値が出力されるファイル。
F_CHR	phase	出力	nfchr.data	電荷密度分布の出力ファイル。デフォルトは、Gaussian cube ファイル形式。
F_DOS	phase ekcal	出力	dos.data	状態密度の出力ファイル。
F_ENERG	ekcal	出力	nfenergy.data	ekcal を用いた計算を行うと、固有値が出力されるファイル。
F_ZAJ	phase ekcal	入出力	zaj.data	波動関数の出力ファイル。計算を継続する際には入力ファイルとなります。バイナリーファイル。
F_CHGT	phase ekcal	入出力	nfchgt.data	電荷密度の出力ファイル。計算を継続する際には入力ファイルとなります。バイナリーファイル。
F_CNTN	phase	入出力	continue.data	ジョブが一旦終了した後、継続するのに必要な情報の出力ファイル。継続計算の際には、このファイルを入力として使います。
F_CNTN_BIN	phase	入出力	continue bin.data	ジョブが一旦終了した後、継続するのに必要な情報の出力ファイル。継続計算の際には、このファイルを入力として使います。バイナリーファイル。
F_CNTN_BIN_PAW	phase ekcal	入出力	contine_bin_paw.data	ジョブが一旦終了した後、継続するのに必要な情報の出力ファイル。PAW を利用している場合、F_CNTN_BIN に加えて必要。継続計算の際には、このファイルを入力として使います。また固定電荷の計算の場合も入力として使います。バイナリーファイル。
F_STATUS	phase ekcal	出力	jobstatus00x	計算状況の出力ファイル。

2.3 計算の実行

2.3.1 プログラム phase の実行

入力パラメータファイル、擬ポテンシャルファイルを実行ディレクトリに置きます。file_names.data を使用する場合には、それも同じディレクトリに置いてください。

1 プロセッサ（1 コア）の逐次計算を行う場合には、次のようにプログラム phase を実行します。“././phase0_2019.01/bin/”は、PHASE のバイナリーが置かれているディレクトリです。

```
% ../../phase0_2019.01/bin/phase
```

ただし、お使いの環境によってはMPI ライブラリの実行コマンドを使用する必要があるかもしれません。

2.3.2 並列計算について

PHASE/0 は、MPI ライブラリにリンクしてコンパイルすることによって並列計算を行うことが可能です。PHASE/0 には「2 次元並列版」と「3 次元並列版」がありますが、前者はバンドと k 点に対する並列実行が、後者はバンド、 k 点に加え波動関数の展開係数に対する並列実行が可能です。3 次元並列版には一部機能制限がありますが、多くの CPU コアが利用できる場合は非常に高速な計算を実行することが可能となっています。

並列計算を行う場合には、お使いの計算機の利用する MPI ライブラリの実行コマンドを使用します。詳細はお使いの計算機システムのマニュアルを参照ください。一般的なコマンドは mpirun です。

2.3.2.1 基本的な並列計算

2 次元版の場合、典型的には以下のようなコマンドになります。

```
% mpirun -np NP PHASE_INSTALL_DIR/phase0_2019.01/bin/phase ne=NE nk=NK
```

ここで、NP は MPI プロセス数、NE はバンド並列数、NK は k 点並列数です。また PHASE_INSTALL_DIR を PHASE/0 をインストールしたディレクトリとしました。NP=NE×NK という関係が満たされている必要があります。 k 点並列の方が効率は良い場合が多いので、 k 点並列は取れるだけ取ることが推奨されます。

3 次元版の場合、典型的には以下のようなコマンドとなります。

```
% mpirun -np NP PHASE_INSTALL_DIR/phase0_2019.01/bin/phase.3d ng=NG ne=NE nk=NK
```

ここで NG が波動関数の展開係数（G ベクトル）の並列数です。NP=NG×NE×NK という関係が満たされている必要があります。多くの場合 NG を NE の 2 倍程度にすると良好なパフォーマンスが得られます。

2.3.2.2 ディレクトリ並列による計算

また、2 次元版、3 次元版共通で「ディレクトリ並列」機能を利用することも可能です。これは、事前に入力ファイルの置かれたディレクトリが複数ある場合にそのディレクトリごとに別個計算を行う機能です。この方法で計算を行うためには、まずは必要な入力の置かれた複数のディレクトリを作成します。つぎに、作成した計算ディレクトリの親ディレクトリに dirlist というファイル名のファイルを作成し、つぎのような内容を記述します。

```
ND
dir1
dir2
...
```

ここで ND がディレクトリ並列の並列数（ディレクトリの数）、dir1, dir2, ... が入力ファイルの置かれたディレクトリの名前です。以上の準備を行った上で、通常通り PHASE を実行します。ただし、2 次元版の場合 NP=ND×NE×NK、3 次元版の場合は NP=ND×NG×NE×NK が満たされるように並列数を調整する必要があります。

ディレクトリ並列機能は主にベリ一位相計算を行う Perl スクリプトが利用することを想定して実装された機能ですが、ベリ一位相計算に限らずあらゆる計算機能で利用することが可能です。ただし、並列化効率自体は非常に高い並列化手法ですが、最も時間のかかるディレクトリが律速になる点にはご注意ください。

2.3.2.3 レプリカ並列による計算

NEB 法および拘束条件付き構造最適化、メタダイナミクス法においては「レプリカ並列」を利用した計算を行うことが可能です。この機能を利用するには、以下のように PHASE/0 を起動します。

2次元版の場合：

```
% mpirun -np NP PHASE_INSTALL_DIR/phase0_2019.01/bin/phase nr=NR ne=NE nk=NK
```

3次元版の場合：

```
% mpirun -np NP PHASE_INSTALL_DIR/phase0_2019.01/bin/phase.3d nr=NR ng=NG ne=NE nk=NK
```

ここで NR はレプリカ並列数です。NP=NR×NG×NE×NK が満たされるように並列数を調整する必要があります。ディレクトリ並列の場合と同様並列化効率自体は非常に高い並列化手法ですが、「最も収束の遅いレプリカ」が律速になる点にご注意ください。

2.3.2.4 k 点数の調べ方

利用可能、もしくは最適な k 点並列数は k 点の数に依存しますが、実際に利用される k 点の数は対称性によって減る場合があるので、事前には分かりません。これを調べるには、まず入力パラメータファイルの `condition` を `preparation` とします。

```
control{  
  condition = preparation  
}
```

このようにして PHASE/0 を実行すると、時間のあまりかからない前処理のみが行われ、計算が終了します。得られたログファイル(output000 ファイル)から k 点の数を調べるには、以下のようなコマンドを発行します。

```
% grep kv3 output000  
!kp kv3 =    70 nspin =    2
```

kv3/nspin が利用可能な k 点並列数の上限となります。がこの例の場合 kv3 = 70, nspin = 2 と出力されていますので、 k 点並列数の上限は 35 となります。また、この上限の数を割り切れる数を採用した方が並列化効率は向上します。なお、実計算を実行する前に `condition` パラメータを適切な値に編集し直すことを忘れないようにしてください。

2.3.3 計算状況の確認

SCF 計算の途中経過は、ログファイル `output000` において、全エネルギーの変遷を見ることによって確認することができます。

ログファイル `output000` において、TOTAL ENERGY FOR で始まる部分に SCF イタレーションごとの全エネルギーが出力されます。

以下の `grep` コマンドを利用することにより、この部分のみを抽出することができます。

```
% grep TH output000
```

```
TOTAL ENERGY FOR 1 -TH ITER= -30.856896066222 edel = -0.308569D+02 : SOLVER = MATDIAGON
TOTAL ENERGY FOR 2 -TH ITER= -31.552303846339 edel = -0.695408D+00 : SOLVER = PDAVIDSON
TOTAL ENERGY FOR 3 -TH ITER= -31.585336745971 edel = -0.330329D-01 : SOLVER = PDAVIDSON
TOTAL ENERGY FOR 4 -TH ITER= -31.587689791426 edel = -0.235305D-02 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR 5 -TH ITER= -31.587917474699 edel = -0.227683D-03 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR 6 -TH ITER= -31.587936742564 edel = -0.192679D-04 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR 7 -TH ITER= -31.587937115320 edel = -0.372756D-06 : SOLVER = SUBMAT + RMM3
.....
.....
```

TOTAL ENERGY FOR ### -TH ITER= の部分に#### 回目の SCF イタレーションにおける全エネルギーがハートリー単位 (Ha) で出力されます (上記の例では、約-31Ha)。edel = のあとには、現在の SCF イタレーションと 1 回前のイタレーションのエネルギー差がハートリー単位で出力されます。この値が、入力パラメータファイルで設定した閾値 (`delta_total_energy`) よりも小さくなると SCF 計算が収束したと見なされます。

SOLVER = のあとには利用したソルバーの情報が出力されます。上記の例では 1 回目が MATDIAGON, 2 回目と 3 回目は P-Davidson, 4 回目以降が部分空間対角化 (SUBMAT) を有効にした RMM3 ソルバーであることが理解できます。このように履歴を調べながら、SCF 計算が収束へ向かっているかどうかを確認することができます。

2.3.4 継続計算

継続して計算を実行することができます。継続計算を実行するには、入力パラメータファイルの `control` ブロックの `condition` を編集します。

```
control{
  condition = continuation
}
```

`condition` に `continuation` を指定すると継続計算を行います。`automatic` を指定すると、継続計算が可能な場合は継続計算、そうでない場合は最初からの計算を行います。`condition` のデフォルト値は `automatic` です。継続計算では直前の計算での出力ファイルの一部を入力ファイルとして使用します。同じ実行ディレクトリで計算を実行する場合には特に配慮することなく、継続計算を実行することが可能です。

2.3.5 ekcal の実行 (状態密度計算、バンド構造計算)

状態密度計算、バンド計算において、`k` 点の取り方を通常の SCF 計算と変更したい場合に使うプログラムとして `ekcal` があります。

SCF 計算の計算結果の電荷密度を入力として計算できます。

2.3.5.1 状態密度の計算

SCF 計算の計算結果の電荷密度ファイル `nfchgt.data` を実行ディレクトリにコピーします。または、入出力ファイル設定ファイル `file_names.data` において、`F_CHG` に SCF 計算の計算結果の電荷密度ファイル指定します。

SCF 計算結果の電荷密度を用いた計算を行うには、入力パラメータファイルの `control` ブロックの `condition` を `fixed_charge` とします。

```
control{
  condition = fixed_charge
}
```

また、`Accuracy` ブロック内に固有値に関する収束条件を指定します。

```
accuracy{
  ek_convergence{
    delta_eigenvalue = 1e-5
  }
}
```

次のようにプログラム `ekcal` を実行します。”`phase0_2019.01/bin/`”は、`PHASE` がインストールされているディレクトリです。

```
% ../../phase0_2019.01/bin/ekcal
```

2.3.5.2 バンド構造計算

SCF 計算の計算結果の電荷密度ファイル `nfchgt.data` を実行ディレクトリにコピーします。または、入出力ファイル設定ファイル `file_names.data` において、`F_CHG` に SCF 計算の計算結果の電荷密度ファイル指定します。

サンプリング **k** 点の設定ファイル `kpoint.data` を用意します。**k** 点の設定ファイル `kpoint.data` は、ツール `band_kpoint.pl` を用いて作成します。以下のようなサンプリング **k** 点の対称点のファイル `bandkpt.in` を用意します。

0.04	k 点の間隔
-0.8333333 0.8333333 0.8333333	
0.8333333 -0.8333333 0.8333333	逆格子ベクトル
0.8333333 0.8333333 -0.8333333	
3 2 1 4 # W	k 点の対称点 n1 n2 n3 nd # Symbol
1 1 1 2 # L	
0 0 0 1 # {/Symbol G}	
1 1 0 2 # X	
3 2 1 4 # W	
5 3 0 8 # K	

各 **k** 点の対象点の指定において、`n1/nd`, `n2/nd`, `n3/nd` が各 **k** 点の対象点の逆格子空間の位置になります。例えば `3 2 1 4 # W` では、逆格子空間の `3/4`, `2/4`, `1/4` の位置に `W` 点を指定しています。

以下のようにツール `band_kpoint.pl` を実行すると、サンプリング **k** 点の設定ファイル `kpoint.data` が作成されます。

```
% ../../phase0_2019.01/tools/bin/band_kpoint.pl bandkpt.in
```

SCF 計算結果の電荷密度を用いた計算を行うには、入力パラメータファイルの `control` ブロックの `condition` を `fixed_charge` とします。

```
control{
  condition = fixed_charge
}
```

`Accuracy` ブロック内に、**k** 点サンプリングと、固有値に関する収束条件を指定します。作成した **k** 点の設定ファ

イルを使用する場合には、`ksampling` の `method` をファイルに指定します。

```
accuracy{
  ksampling{
    method = file
  }
  ek_convergence{
    delta_eigenvalue = 1e-5
  }
}
```

次のようにプログラム `ekcal` を実行します。”`phase0_2019.01/bin/`”は、PHASE がインストールされているディレクトリです。

```
% ../../phase0_2019.01/bin/ekcal
```

2.3.5.3 3D 版について

3D 版には、`ekcal` プログラムは付属しません。そのかわり、PHASE を利用して同じ計算を実行することができます。固定電荷計算を PHASE で行くと、デフォルトの状態ではすべての **k** 点を一括で処理するモードで動作します。`ekcal` のように **k** 点を一点ずつ処理する場合、入力パラメーターファイルに以下のような記述を行います。

```
control{
  condition = fixed_charge
  fixed_charge_option{
    kparallel = one_by_one
  }
}
```

いずれのモードでもバンド、**k** 点、**G** 点並列を組み合わせることができます。ただし、ベリ一位相計算の場合は上述の `kparallel` を `one_by_one` とし、また **k** 点並列を使わないようにしてください。

2.4 計算結果の確認

2.4.1 計算終了の状況・要因・対応

計算終了の状況・要因・対応を以下にまとめます。

計算状況	プログラム実行の終了の要因	対応
正常終了 SCF 計算が収束 (構造緩和計算が終了)	SCF 計算の収束判定条件を満たした場合 ・全エネルギー変化量が閾値より小さくなった場合 構造緩和計算の収束判定条件を満たした場合 ・原子に働く力の最大値が閾値より小さくなった場合	計算終了 解析計算を行う
正常終了 SCF 計算が未収束	SCF 計算の最大イタレーション数に達した場合 ・ SCF 計算のイタレーション数が、control ブロックの <code>max_iteration</code> の値を超えた場合 ファイル <code>nfstop.data</code> で指定したイタレーション数を超えた場合 ・デフォルトの設定では、1 イタレーションおきに、 <code>nfstop.data</code> ファイルを読み込みます。この方法を使うことにより、計算開始後でも、任意のタイミングで計算を終了させることができます。 計算時間の最大時間に達した場合 ・ Control ブロックの <code>cpumax</code> を計算時間が超えた場合	継続計算を行う
異常終了	計算機トラブル 入力パラメータファイルの記述ミス 擬ポテンシャルファイルが存在しない (プログラム内部の問題)	入力データを見直して再計算

2.4.2 計算終了（正常終了、異常終了）の確認

PHASE の実行が正常に終了すると、ログファイル (output000 ファイル) の最後に以下のような情報が出力されます。

..... <<Total elapsed CPU Time until now = 81.69520 (sec.)>> closed filenumber = 31 closed filenumber = 52 closed filenumber = 53 closed filenumber = 55 closed filenumber = 42 closed filenumber = 43 closed filenumber = 44 closed filenumber = 75 closed filenumber = 65 closed filenumber = 66
--

Total elapsed CPU Time until now =のあとに計算時間が出力されます。ログファイルの最後がこのようになっていなければ PHASE は正常終了していません。なんらかの要因のため異常終了しています。その場合は、入力ファイルなどを変更して再計算してください。

2.4.3 SCF 計算、構造緩和計算の収束状況の確認

PHASE の計算が正常に終了した場合でも、ユーザーが必要とする計算が終了したことにはなりません。PHASE がどのような状況で終了したかは、計算終了後に出力される「continue.data」というファイルの最後から8行目（網掛け部分）の「convergence」の下に出力されている数によって収束状態を知ることが出来ます。2 の場合には、SCF 計算が無事収束した後、力の計算を行い、その力の大きさが入力パラメータファイルで指定した力の閾値（delta_total_energy）より小さくなっていることを意味します。つまり、構造緩和計算が終了したことを意味します。2 以外の場合には構造緩和計算が終了していないため、継続計算をしてください。

```
iteration, iteration_ionic, iteration_electronic
      11           1           11
Ionic System
(natm)
      2
(pos)
0.1250000000000000D+00 0.1250000000000000D+00 0.1250000000000000D+00
-0.1250000000000000D+00 -0.1250000000000000D+00 -0.1250000000000000D+00
(cps)
0.1290824363824501D+01 0.1290824363824501D+01 0.1290824363824501D+01
-0.1290824363824501D+01 -0.1290824363824501D+01 -0.1290824363824501D+01
(cpd)
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00
(cpo( 1))
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00
(cpo( 2))
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00
(cpo( 3))
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00
forcmx_constraint_quench
0.1000000000000000D+03
Total Energy
-0.7878566524513241D+01 -0.7878566524513241D+01
isolver
      5
convergence
      2
edelta_ontheway
0.1000000000000000D-09
corecharge_cntnbin
      0
neg
      8
```

2.4.4 計算状況の確認（ログファイル output000 および jobstatus000）

計算実行のログファイルは output000 というファイル名で出力されます。000 の部分は同じディレクトリにおいて計算を実施した回数に応じて 001, 002, ... と増えていきます。

このファイルには、様々な情報、物理量が出力されます。中でも参照することの多い情報について説明します。

2.4.4.1 サンプリング k 点

k 点是对称性を考慮している場合は入力の設定から予想しづらい場合があります。そこで、最終的に利用する k 点の数を調べたい場合があります。これは、ログファイル中で kv3 という文字列を検索すると調べられます。

```
!kp kv3 =      8 nspin =      1
```

kv3 =のあとの8が k 点の数です。nspin =のあとの1は、スピン自由度を考慮していないことに相当します。スピン自由度を考慮している場合、この数値は2になります。

2.4.4.2 全エネルギー

全エネルギーは、ログファイルに以下のように出力されます。

```
TOTAL ENERGY FOR      3 -TH ITER=   -687.253021587082 edel =   -0.215950D+02 : SOLVER =P DAVIDSO
N
KI=      294.118626755617 HA=      4820.263454482710 XC=      -686.596385560733 LO=      -8452.905431759591
NL=     -349.620400894588 EW=      3182.022578317359 PC=      505.464805336868 EN=      -0.000268264724
PHYSICALLY CORRECT ENERGY =     -687.252887454720
```

TOTAL ENERGY FOR ... に全エネルギーが、edel =のあとに1回前のイタレーションとの差が出力されます。次の行とその次の行には全エネルギーを構成する各項のエネルギーの値が記録されます。KI は運動エネルギー、HA はHartree エネルギー、XCは相関交換エネルギー、LO は局所エネルギー、NL は非局所エネルギー、EW はEwald エネルギー、PC はコア補正エネルギー、EN はエントロピーを表します。これらの総和が全エネルギーになります。

PHYSICALLY CORRECT ENERGY のあとには、スメアリングを行っている場合に“0 K へ補外した全エネルギー” が記録されます。

2.4.4.3 スピン状態

スピンを考慮した計算を実行している場合、下記のように各 SCF イタレーションにおける多数派スピン状態および少数派スピン状態の数が記録されます。

```
!OLD total charge (UP, DOWN, SUM) =      4.53623488 (+)      3.46376512 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.64907433 (+)      3.35092567 (=)      8.00000000
```

!OLD から始まる行には1回前のイタレーションにおける電荷の情報が、!NEW から始まる行には現在のイタレーションの電荷の情報が出力されます

2.4.4.4 固有値および占有数

各 k 点におけるフェルミエネルギー近傍の固有値は、計算終了直前に以下のように出力されます。

```
EFermi =      0.10922262
===== Energy Eigen Values in the vicinity of the Fermi energy level (Range=      1 :      12) =====
ik =      1      -0.198857      0.025583      0.025583      0.025583      0.073053      0.073053
           1.213861      1.213861      1.299925      1.299925      1.299925      1.636396
ik =      2      -0.196917      0.104776      0.104776      0.104776      0.174253      0.174253
           1.217079      1.217079      1.325796      1.325796      1.325796      1.689134
ik =      3      -0.191252      0.025548      0.025548      0.027605      0.069739      0.074378
           0.983941      1.019452      1.019452      1.095653      1.131322      1.131322
ik =      4      -0.189306      0.104412      0.104412      0.107058      0.169882      0.175940
           0.977036      1.012673      1.012673      1.124796      1.174308      1.174308
ik =      5      -0.169140      0.025903      0.025903      0.033499      0.060085      0.078167
           0.910397      0.964244      0.964244      1.009529      1.125585      1.125585
ik =      6      -0.167035      0.103921      0.103921      0.113720      0.157186      0.180510
           0.910306      0.977222      0.977222      1.034555      1.149955      1.149955
.....
.....
.....
```

続けて、占有数が次のように出力されます。

```
===== Occupations in the vicinity of the Fermi energy level (Range=      1 :      12) =====
ik =      1      1.000000      1.000000      1.000000      1.000000      1.000000      1.000000
           0.000000      0.000000      0.000000      0.000000      0.000000      0.000000
ik =      2      1.000000      0.884668      0.884668      0.884668      0.000000      0.000000
           0.000000      0.000000      0.000000      0.000000      0.000000      0.000000
ik =      3      1.000000      1.000000      1.000000      1.000000      1.000000      1.000000
```

		0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
ik =	4	1.000000	1.000000	1.000000	0.261115	0.000000	0.000000
		0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
ik =	5	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
		0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
ik =	6	1.000000	1.000000	1.000000	0.000995	0.000000	0.000000
		0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
.....							
.....							
.....							

占有数は一般には0～1の間の値をとるので、例えば、スピンを考えない計算の場合、1.0は電子2個に占有されることを表します。対称性によりk点の縮約がある場合には、占有数は縮約に応じて変わりますので、その点にも注意する必要があります（バルクの計算のように小さい系でk点数を多数取る場合にこのようなことが起こります）。

2.4.4.5 SCF 計算あたりの計算時間

計算時間は、`printoutlevel` の `base` が 1 以上の場合、SCF イタレーションごとに以下のように出力されます。

<< CPU Time Consumption -- TOP 9 Subroutines (2) >>						
no	id	subroutine name	time(sec)	r(%)	count	no(2)
1	20	evolve_WFs_in_subspace (davidson	115.74820	71.17	8	1
2	13	m_ES_Vnonlocal_W	10.78620	6.63	8	2
3	8	betar_dot_WFs	7.33490	4.51	14	3
4	16	m_CD_softpart	2.53880	1.56	1	4
5	7	m_XC_cal_potential	0.97520	0.60	2	5
6	17	m_CD_hardpart	0.28100	0.17	1	6
7	10	m_ES_Vlocal_in_Rspace	0.02990	0.02	1	7
8	19	m_CD_mix_pulay	0.00670	0.00	1	8
9	18	m_CD_convergence_check	0.00230	0.00	1	9
Total cputime of (2)-th iteration			162.64080 /	221.651 (sec.)		

上記のように、各ルーチンにおける計算時間が出力されます。この情報は、1つ前のイタレーションとの時間の差が5%以下の場合には出力されません。

2.4.4.6 計算の進捗状況 (jobstatus000)

`jobstatus000` ファイルは、計算の進捗をまとめたファイルです。000の部分には、`output000`と同様に計算を繰り返し行くと001, 002, ... となります。その内容は、以下のようになります。

status	=	FINISHED
iteration	=	674
iter_ionic	=	21
iter_elec	=	23
elapsed_time	=	51648.7582

`status` FINISHED (計算終了), ITERATIVE (繰り返し計算中), START (初期化中)
`iteration` 電子状態の繰り返し計算回数が表示されます。
`iter_ionic` 原子座標の更新回数が表示されます。
`iter_elec` 現在の原子配置における電子状態の繰り返し計算回数が表示されます。
`elapsed_time` 経過時間が秒単位で表示されます。

2.5 計算結果の解析、可視化

2.5.1 全エネルギー、原子に働く力の最大値（エネルギー履歴ファイル nfefn.data）

ファイル `nfefn.data`（または `file_names.data` ファイルにおいて `F_ENF` によって指定されるファイル）には、系の全エネルギーや原子に働く力の最大値、さらに分子動力学シミュレーションを行った場合はイオンの運動エネルギーや保存量なども記述されます。

構造緩和を行った場合と分子動力学シミュレーションを行った場合とで出力内容が異なるので、それぞれについて説明します。

2.5.1.1 構造緩和計算

典型的な構造緩和を行った後の `nfefn.data` の例を示します。

iter_ion	iter_total	etotal	forcmx
1	24	-108.4397629733	0.0086160410
2	40	-108.4401764388	0.0076051917
3	56	-108.4405310817	0.0068758156
4	73	-108.4410640011	0.0065717365
5	94	-108.4414256084	0.0099533097
6	113	-108.4414317178	0.0094159378
		
		
		

各列は各々次のような量に対応します。

<code>iter_ion</code>	イオンの更新回数です。
<code>iter_total</code>	SCF ループの更新回数です。この数字は通算の値が記述されます。
<code>etotal</code>	全エネルギーを、ハートリー単位で出力します。
<code>forcmx</code>	原子に働く力の最大値を原子単位(hartree/bohr ³)で記述します。この値が入力ファイルにて与えた構造緩和の収束判定を満たすまで計算は実行されます。

2.5.1.2 分子動力学法計算

分子動力学法計算の場合、下記のようになります。

iter_ion	iter_total	etotal	ekina	econst	forcmx
1	18	-7.8953179624	0.0000000000	-7.8953179624	0.0186964345
2	30	-7.8953851218	0.0000665502	-7.8953185716	0.0183575425
3	43	-7.8955768901	0.0002565396	-7.8953203505	0.0173392067
				
				
				

構造緩和の場合とほぼ同様ですが、新たな列が追加されます。

<code>ekina</code>	系の運動エネルギー、
<code>econst</code>	系の保存量、すなわちエネルギー一定の分子動力学シミュレーションの場合系の全エネルギー、温度一定の分子動力学シミュレーションの場合系の全エネルギーに熱浴のエネルギーを加えた量です。

2.5.1.3 格子最適化計算

格子の最適化を行った場合、下記のようになります。

iter	unitcell	iter_ion	iter_total	etotal	forcmx	stressmx
1	1	25	-181.4043211381	0.0019960638		
1	2	33	-181.4043560304	0.0004826299		
1	3	40	-181.4043582176	0.0000016495		0.0002327496
2	1	49	-181.4044223602	0.0000572790		0.0002273231
3	1	58	-181.4044833189	0.0001158383		0.0002220365
					
					
					

通常の構造最適化のケースに加え、以下の列が加えられます。

iter_unitcell	格子の更新回数
stressmx	ストレステンソルの最大値

2.5.2 原子座標 (原子座標履歴ファイル nfdynm.data)

ファイル `nfdynm.data` (または `file_names.data` ファイルにおいて `F_DYNM` によって指定されるファイル) には、各原子の座標とそれに働く力が記述されます。構造緩和や分子動力学シミュレーションを行った場合はイオンの更新の回数分だけデータが書き込まれます。典型的な `nfdynm.data` の中身を以下に記述します。なお、このファイルにおいて利用される単位系はすべて原子単位系です。

```

#
#   a_vector =          9.2863024980          0.0000000000          0.0000000000
#   b_vector =         -4.6431512490          8.0421738710          0.0000000000          (a)
#   c_vector =          0.0000000000          0.0000000000          10.2158587136
#   ntyp =          2 natm =          9          (b)
# (natm->type)      1      1      1      1      1      1      2      2      2          (c)
# (speciesname)    1 :      O          (d)
# (speciesname)    2 :      Si
#
cps and forc at (iter_ion, iter_total =          1          24 )          (e)
1      3.161057370      1.169332082      1.214972077      -0.004058      -0.005565      -0.004966          (f)
2      6.693102525      2.152889944      4.620258315          0.006945      -0.001028      -0.004994
3      4.075293851      4.719951845      8.025544553      -0.002872          0.006394      -0.004796
4      -1.482093879      6.872841789      5.595600399      -0.004362          0.005502          0.004993
5      -0.567857398      3.322222026      9.000886637      -0.002792      -0.006296          0.004965
6      2.049951276      5.889283925      2.190314161          0.006974          0.000708          0.004795
7      4.921740324          0.000000000      3.405282833          0.001436          0.000122          0.000068
8      -2.460870162      4.262352150      6.810569070      -0.000612          0.001305      -0.000066
9      2.182281087      3.779821719      10.215855308      -0.000660      -0.001143          0.000001
cps and forc at (iter_ion, iter_total =          2          40 )
1      3.156999743      1.163767576      1.210005993      -0.002904      -0.005755      -0.003892
2      6.700048015      2.151861938      4.615264365          0.006567          0.000186      -0.003832
3      4.072421499      4.726345880      8.020748072      -0.003503          0.005487      -0.003829
4      -1.486455954      6.878343743      5.600593135      -0.003122          0.005780          0.003831
5      -0.570648922      3.315925959      9.005851266      -0.003532      -0.005392          0.003892
6      2.056925355      5.889992076      2.195109289          0.006503      -0.000290          0.003828
7      4.923176344          0.000121757      3.405351146          0.000397      -0.000013          0.000018
8      -2.461482612      4.263656762      6.810503226      -0.000210          0.000337      -0.000017
9      2.181621403      3.778679157      10.215856638      -0.000197      -0.000341          0.000000
          .....
          .....
          .....
          .....
          .....

```

- (a) セルベクトルが書かれています。 `a_vector`, `b_vector`, `c_vector` にそれぞれ a 軸, b 軸, c 軸のベクトルが記述されています。
- (b) `ntyp =` の後には使用されている原子種の数記述されています。この例では 2 です。また, `natm =` の後には原子数が書かれています。この例では 9 です。
- (c) (`natom→type`) の後には, 原子と原子種のマッピングが書かれています。この例だと, 1 番目から 6 番目の原子の原子種は 1, 7 番目から 9 番目の原子種は 2 という元素に対応します。
- (d) (`speciesname`) の後には, 原子種と ID のマッピングが書かれています。この例では, 1 という原子種は O(酸素), 2 という原子種は Si(珪素) に相当する, ということになります。
- (e) 各ステップでの情報が記述されています。この例では, イオンの更新回数が 1 回, SCF の更新回数が 24 回となります。
- (f) 実際の原子の場所とその原子に働いている力が記述されています。1 番目の列は原子の ID, 2 番目から 4 番目の列が原子の場所の `x,y,z` 座標, 5 番目から 6 番目の列が原子に働く力の `x,y,z` 座標となります。もし, 入力ファイルにおいて `printlevel` ブロックの `velocity` 変数を 2 に設定していた場合, 7 番目から 9 番目の列に速度が原子単位で出力されます。

2.5.3 電荷密度 (電荷密度ファイル nfchr.cube)

ファイル nfchr.cube (または file_names.data ファイルにおいて F_CHR によって指定されるファイル) には、Gaussian CUBE 形式の電荷密度分布が出力されます。

PHASE Viewer や、Gaussian CUBE 形式の可視化に対応している可視化ソフトウェアを利用して、原子構造、電荷密度を可視化してください。

2.5.4 状態密度 (状態密度ファイル dos.data)

ファイル dos.data (または file_names.data ファイルにおいて F_DOS によって指定されるファイル) には、状態密度が出力されます。

状態密度のグラフを作成するには、次のようにツール dos.pl を実行します。”phase0_2019.01/bin/”は、PHASE がインストールされているディレクトリです。ツール dos.pl は、Perl スクリプトです。実行後、状態密度のグラフの EPS 形式の画像ファイル density_of_states.eps が作成されます。

```
% ../../phase0_2019.01/tools/bin/dos.pl dos.data -erange=-15,10 -with_fermi -color
```

dos.data	状態密度の出力ファイル
-erange	表示するエネルギーの範囲を指定します。-15,10 は-15eV から 10eV の範囲を指定しています。
-with_fermi	フェルミエネルギーを表示します。
-color	カラー出力を行います。

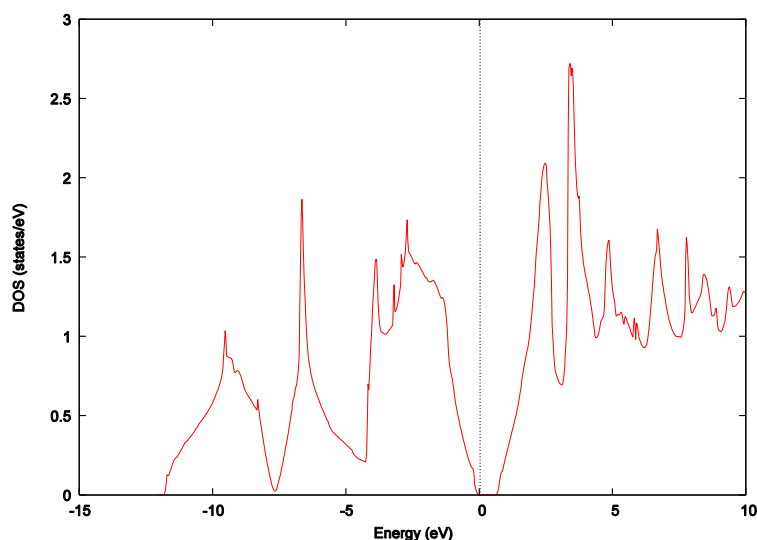


図 2-2 状態密度 可視化例 (Si ダイヤモンド構造のバンド構造)

2.5.5 バンド構造 (固有値データファイル nfenergy.data)

ファイル `nfenergy.data` (または `file_names.data` ファイルにおいて `F_ENERG` によって指定されるファイル) には、サンプリング k 点における固有値が出力されます。

バンド構造のグラフを作成するには、次のようにツール `band.pl` を実行します。`”phase0_2019.01/bin/”`は、PHASE がインストールされているディレクトリです。ツール `dos.pl` は、Perl スクリプトです。実行後、状態密度のグラフの EPS 形式の画像ファイル `band_structure.eps` が作成されます。

```
% ../../phase0_2019.01/tools/bin/ band.pl nfenergy.data bandkpt.in -erange=-15,10 -with_fermi -color
```

<code>nfenergy.data</code>	サンプリング k 点における固有値の出力ファイル
<code>bandkpt.in</code>	サンプリング k 点に関するファイル
<code>-erange</code>	表示するエネルギーの範囲を指定します。-15,10 は-15eV から 10eV の範囲を指定しています。
<code>-with_fermi</code>	フェルミエネルギーを表示します。
<code>-color</code>	カラー出力を行います。

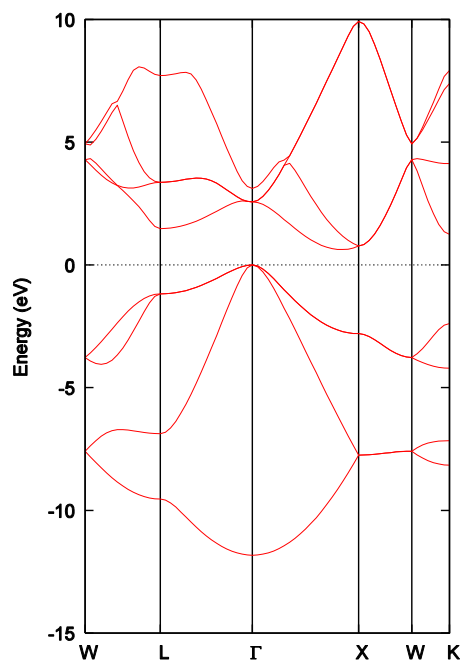


図 2-3 バンド構造 可視化例 (Si ダイヤモンド構造のバンド構造)

2.6 参考文献

- [1] H. J. Monkhorst and J. D. Pack, Phys. Rev. B 13, 5188 (1976).
- [2] P. Pulay, Chem. Phys. Lett. 73, 393 (1980).
- [3] C.G. Broyden, Math. Comput. 19, 577, (1965).
- [4] P.E. Blöchl, Phys. Rev. B 50, 17953 (1994).
- [5] N. Troullier and J.L. Martins, Phys. Rev. B **43**, 1993 (1991).
- [6] D. Vanderbilt, Phys. Rev. B **41** 7892 (1990).
- [7] G. Kresse and D. Joubert, Phys. Rev. B **59**, 1758, (1999).
- [8] J. P. Perdew, K. Burke, and M. Ernzerhof, Phys. Rev. Lett. **77**, 3865 (1996).
- [9] J. P. Perdew and Y. Wang, Phys. Rev. B **45**, 13244 (1992).

3. 入力パラメータファイル : nfinp.data (詳細版)

3.1 入力パラメータファイルの形式(F_INP ファイル)

入力パラメータファイル **nfinp.data** は、どのようなモデル（原子配置など）に対し、どのような条件で計算するかという情報を記述するファイルです。

デフォルトのファイル名は **nfinp.data** ですが、**file_names.data** において、**F_INP** キーワードを使って自由に名前を指定できます。例えば、計算する系に関連した名前をつけることも可能です。

3.1.1 パラメータ設定形式

このファイルは、タグ名と中括弧`{}`で囲まれたブロックの階層構造で記述します。計算パラメータの設定は、タグ形式になっており、各タグに、結晶構造、計算精度、計算の制御などの情報を記入します。以下に、入力パラメータファイルの記述方法を簡単に説明します。

関連のある入力データはまとめて一つの「ブロック」内に記述します。ブロックは、`ブロック名{...}` という形式で階層的に記述します。たとえば、以下のようになります。基本的に計算パラメータは、`タグ=値`の形で指定します。ただし、パラメータごとに指定形式が異なります。各計算パラメータの指定方法の説明を参照ください。

```
Upper_block{
  Lower_block{
    ...
    tag_keyword = value
  }
}
```

入力パラメータファイルの作成・編集において、以下の点に注意ください。

- 同じ階層に同じ名前のブロックを記述することはできません。
- ブロック名の太文字・小文字を区別することはありません。
- ブロック名が間違っている場合には、そのブロック全体が記述されていないものとみなされます（無視されます）。その場合は、デフォルト値が採用されます。エラーメッセージは表示されません。
- 変数は、改行で区切るほかカンマ(,) で区切ることもできます。
- 文字列型の変数に空白文字を含めたい場合、半角の 2 重引用符(") で囲みます。
- 全角文字は使用しないでください。

最上位のブロックは、以下のものがあります。

control ブロック	全体的な計算条件の設定
accracy ブロック	計算精度の設定
structure ブロック	原子構造の設定
wavefunction_solver ブロック	波動関数ソルバーの設定
charge_mixing ブロック	電荷密度混合法の設定
structure_evolution ブロック	構造最適化、分子動力学法計算の設定
postproccesing ブロック	後処理の設定
printlevel ブロック	ログ出力の設定

3.1.2 単位の指定

PHASE の入力ファイルのデフォルトの単位は原子単位ですが、単位を明示的に指定することも可能です。表 3.1 の単位を利用することができます(デフォルトの単位は太字で表示されています)。

表 3.1 PHASE で利用可能な単位

長さ	bohr , angstrom, nm
エネルギー	hartree , eV, rydberg
時間	au_time , fs, ps, ns, s, sec, min, hour, day
速度	bohr/au_time , bohr/fs, angstrom/fs, angstrom/au_time, nm/fs, nm/au_time
力	hartree/bohr , hartree/angstrom, hartree/nm, eV/angstrom, eV/bohr, ev/nm, rydberg/bohr, rydberg/angstrom, rydberg/nm
圧力	hartree/bohr3 , hartree/angstrom3, hartree/nm3, eV/angstrom3, eV/bohr3, eV/nm3, rydberg/angstrom3, rydberg/bohr3, rydberg/nm3,
質量	au_mass , atomic_mass,

単位は、実数型のデータに直接指定する方法だけでなく、ブロック単位のデフォルト値を指定することもできます。ブロック単位でデフォルトの単位を指定するには、次のように記述します。

```
block{
  #units angstrom
  ...
  ...
}
```

この例では、**block** の長さの単位のデフォルトがÅ単位になります。複数指定する場合（長さ、エネルギーなど）、空白で区切って指定してください。

3.1.3 コメント

!または//ではじまる行は、コメント扱いとなります。

```
block{
! comment
! tag_keyword = value1      コメント
// tag_keyword = value2     コメント
  tag_keyword = value3
}
```

ただし、**#**と、**!**のあとに**#**が続く場合はコメントとはみなされないので注意してください。

3.1.4 入力パラメータファイル例

Si ダイヤモンド結晶 (2 原子) の電子状態計算を行う場合の基本的な計算条件を記述した入力ファイル例です。

```
control{
  condition = initial
  cpumax = 86400 sec
  max_iteration = 10000
}

accuracy{
  cutoff_wf = 25.0 rydberg
  cutoff_cd = 100.0 rydberg
  num_bands = 8
  ksampling{
    method = monk
    mesh{
      nx = 10
      ny = 10
      nz = 10
    }
  }
  initial_wavefunctions = atomic_orbitals
  initial_charge_density = atomic_charge_density
  scf_convergence{
    delta_total_energy = 1e-10
    succession = 3
  }
  force_convergence{
    max_force = 0.001 hartree/bohr
  }
}

structure{
  element_list{
    #tag    element    atomicnumber
      Si     14
  }
  unit_cell{
    #units angstrom
    a_vector = 0 2.732299538 2.732299538
    b_vector = 2.732299538 0 2.732299538
    c_vector = 2.732299538 2.732299538 0
  }
  unit_cell_type = bravais
  atom_list{
    atoms{
      #tag    element    rx    ry    rz    mobile
      Si     0.125  0.125  0.125    0
      Si    -0.125 -0.125 -0.125    0
    }
    coordinate_system = internal
  }
}

wavefunction_solver{
  solvers{
    #tag    sol    till_n    prec    cmix    submat
  }
}
```

```

        davidson 1 on 1 on
        rmm3 -1 on 1 on
    }
    rmm{
        edelta_change_to_rmm=5e-5
    }
}

charge_mixing{
    mixing_methods{
        #tag no method rmxs rmxe istr prec nbmix
        1 pulay 0.40 0.40 3 on 15
    }
}

Postprocessing{
    dos{
        sw_dos = ON
        deltaE = 1.e-4 hartree
    }
    charge{
        sw_charge_rspace = ON
        filetype = cube !{cube|density_only}
        title = "This is a title line for the bulk Si"
    }
}

```

3.2 入力パラメータファイル nfinp.data のタグ（キーワード）の一覧

入力パラメータファイル nfinp.data のタグ（キーワード）の一覧を、表 3.2 に示します。

表 3.2 入力パラメータファイル nfinp.data のタグ（キーワード）の一覧

最上位ブロック	第 2, 3 ブロック	タグ（キーワード）	説明
control			全体的な計算条件設定ブロック
		condition	preparation, -2 : 入力座標の表示, 対称操作の生成, k 点の生成までで終了します。 automatic, -1 : 継続可能であれば, 計算継続になります。そうでなければ, 計算開始になります。 initial, 0 : 計算開始 continuation, 1 : 計算継続 (以下の 2 つは ekcal による計算で使用) fixed_charge, 2 : 電荷を固定して計算 fixed_charge_continuation, 3 : 固定電荷+計算継続 デフォルト値は automatic です。
		cpumax	CPU 時間の上限 (デフォルト : 86400 sec) 単位 : {sec, min, hour, day}
		max_iteration max_total_scf_iteration	総 SCF 回数 (イタレーション) の制限値 (デフォルト : 10000)
		max_mdstep	総 MD 計算数の制限値 (デフォルトは無制限)
		max_scf_iteration	1MD ステップ内の SCF 回数の制限値 (デフォルトは無制限)
		nfstopcheck	ファイル nfstop.data に書かれた数値で, 処理を停止すべき更新回数を決定 (デフォルト : 1)
		sw_ekzaj	phase で, ekcal の入力となる波動関数ファイル F_ZAJ への出力を行うとき ON にする。 EKCAL でそのファイルを読み込むときも ON にする。ただし, Γ 点の計算でしか使用できません。 デフォルト値は OFF です。
accuracy			計算精度の制御用ブロック識別子
		cutoff_wf	波動関数のカットオフエネルギー
		cutoff_cd	電荷密度のカットオフエネルギー
		num_bands	バンド数
	ksampling		
		method	k 点のサンプリング法。 monk : Monkhorst-Pack 法。 mesh : メッシュを生成します。 file : ファイルから入力 direct_in : 直接記述 gamma : Γ 点のみ デフォルトは Monkhorst-Pack 法
	mesh		メッシュの分割数
		nx, ny, nz	x,y,z 方向への分割数 デフォルト値=(4,4,4), 上限値=(20,20,20)

	kshift		Monkhorst-Pack 法でのみ有効なタグ
		k1, k2, k3	メッシュのずれの指定 (入力値は[0.0, 0.5] の範囲) デフォルト値: hexagonal の場合: k1 = k2 = 0, k3 = 0.5 それ以外の場合: k1 = k2 = k3 = 0.5 ただし 0.5 はメッシュの刻み幅の半分の値を指す
	kpoints		k 点の重みづけ
		kx ky kz denom weight	$\vec{k} = (kx/denom, ky/denom, kz/denom)$ k 点の座標値と, その重みづけ
	smearing		k 点サンプリングの smearing
		method	parabolic : Parabolic 法 (デフォルト) cold : Cold smearing 法(金属系で有効) tetrahedron : Tetrahedron 法 improved_tetrahedron : 改良 tetrahedron 法 tetrahedron または improved_tetrahedron としたときには k 点のサンプリングをメッシュ法にしなければなりません。
		width	smearing 幅 (デフォルト値 : 0.001 hartree) method = parabolic と cold の時に使用
	(タグなし)		
		xctype	交換相関エネルギー(LDA, GGA) LDA : LDAPW91, PZ GGA : GGAPBE, REVPBE
	scf_convergence		自己無撞着場の収束判定
		delta_total_energy	原子あたりの全エネルギーの計算誤差の上限 ΔE (デフォルト値 : 10^{-9} hartree)
		succession	誤差が ΔE 未満に何回連続して収まった時に計算を停止させるかを指定する回数 (デフォルト値 : 2)
	force_convergence		力の収束判定
		max_force	全原子に働く力の最大値がこの値より小さくなれば計算を停止 (デフォルト値 : 0.001 hartree/bohr)
	ek_convergence		固有値の収束判定。ekcal による計算専用の識別子
		num_extra_bands	未収束でもかまわない追加のバンドの数 (デフォルト値 : 2)
		num_max_iteration	k 点一個当たりの最大の更新回数 (デフォルト値 : 300)
		sw_eval_eig_diff	固有値評価用スイッチ {1, on, yes} : 評価あり (デフォルト) {0, off, no} : 評価なし
		delta_eigenvalue	固有値の許容誤差 (デフォルト値 : 10^{-5} hartree)
		succession	計算の繰り返し回数 (デフォルト値 : 2)
	(タグなし)		
		initial_wavefunctions	波動関数の初期値 選択肢 : {random_numbers, matrix_diagonal, atomic_orbitals, file}

			random_numbers:乱数で初期化 matrix_diagon:小行列対角化で初期化 atomic_orbitals:原子軌道で初期化 file:ファイル F_ZAJ から入力
	matrix_diagon		波動関数の初期値を小行列対角化法で与える
		cutoff_wf	波動関数のカットオフ
	(タグなし)		
		initial_charge_density	電荷密度の初期値 選択肢: {Gauss, atomic_charge_density, file} Gauss: ガウス分布関数の重ね合わせで初期化 atomic_charge_density: 原子の電子密度の重ね合わせで初期化 file:ファイル F_CHGT から入力
	precalculation		
		nel_Ylm	予め計算してメモリー上に保持しておく球面調和関数の最高次数 (デフォルト値は 9)
structure			構造設定用ブロック識別子
	unit_cell_type		単位胞の型。選択肢: {primitive, Bravais}
	unit_cell	a_vector b_vector c_vector a, b, c alpha, beta, gamma	unit cell 単位胞の指定。以下のいずれかの方法で与える 各格子ベクトルの (x, y, z) 成分 デフォルトの単位は Bohr 格子定数 a, b, c b-c 軸, c-a 軸, a-b 軸のなす角 (角度のデフォルト単位は degree)
	symmetry		
		method	選択肢:{manual, automatic} automatic を選択すると自動的に対称性を決定します。
		crystal_structure	選択肢: {diamond, hexagonal, fcc, bcc, simple cubic}
	tSPACE		柳瀬章「空間群のプログラム TSPACE」(裳華房) および, ABCAP のマニュアルを参照
		lattice_system	{rhombohedral, trigonal, r, t, -1} {hexagonal, h, 0}, {primitive, simple, p, s, 1} {facecentered, f, 2}, {bodycentered, b, 3} {bottomcentered, basecentered, onefacecentered, bot, ba, o, 4}
		num_generators	生成元の数(1 ~ 3の整数値)
		generators	生成元
		af_generator	磁性空間群の生成元
	(第 3 タグなし)		
		sw_inversion	反転対称性の有無
	(第 2 タグなし)		
		magnetic_state	入力値: {para, antiferro, ferro} から選択 antiferro は af と省略も可
	atom_list		原子構成
		coordinate_system	選択肢: {cartesian, internal}
	atoms		

		rx, ry, rz	座標
		element	元素名
		mobile	可動性 入力値は{1,0}, {on,off}, {yes,no} のどれでも可
		weight	重みづけ weight = 2 は sw_inversion = on の時のみ有効 このとき、反転対称の位置にも原子を生成
	element_list		
		element	元素名(atoms の element の入力値と一致させる)
		atomic_number	原子番号
		mass	質量
		zeta	スピン分極: $\zeta = (n_{\uparrow} - n_{\downarrow}) / (n_{\uparrow} + n_{\downarrow})$
		deviation	初期電荷をガウス関数の和で与えるときの各ガウス関数の偏差。 $\rho(r) = A \exp(-r^2/2\sigma^2)$ における原子の電子密度分布の広がりを決めるパラメータ σ 。 タグ名には dev や standard_deviation も使用可
wavefunction_solvers	solver		波動関数ソルバー (詳しくは 3.6 と 3.8 を参照)
		sol	ソルバーの種類 MatrixDiagon: 行列対角化法 lm+MSD: lm(一次元探索)+ MSD(改良型最急降下法) RMM2P, RMM3: RMM 法 MSD: 修正最速降下法 pdavidson: 分割 Davidson 法 pkosugi: 分割 Kosugi 法
		till_n	何回の更新まで, sol で指定された波動関数の更新方法を適用するかを指定
		dts	計算開始時の時間刻み幅
		dte	itr で指定された更新の回数における時間刻み幅。dts の値のみが入力された場合には dte にも同じ値を適用。
		itr	時間刻み幅を変化させる回数の指定
		var	補間の形式。選択肢: {linear, tanh}
		prec	前処理の有無。選択肢: {on,off}
		cmix	電荷混合法の指定用変数。charge_mixing タグの mixing_methods で指定されている, 各方法に割り振られた番号を使って指定する。
		submat	on のとき subspace_rotation の指定に従って subspace rotation を行う。選択肢: {on,off}
	line_minimization		一次元探索に関する制御
		dt_lower_critical dt_upper_critical	一次元探索をおこなう時の時間刻みの下限と上限 (デフォルト値はそれぞれ, 0.005 と 2.0)
		delta_lmdenom	
	rmm		残差最小化法
		imGSrmm	RMM 法で更新した波動関数に対して, Gram-Schmidt の直交化法を適用する頻度

			(デフォルト値は、 毎回実行の <code>imGSrmm = 1</code>)
		<code>rr_Critical_Value</code>	バンド毎の収束判定条件。波動関数の残差のノルムがここで指定された値以下になれば、そのバンドはそれ以降更新されない
		<code>edelta_change_to_rmm</code>	波動関数のソルバーを RMM 法に変えるときの、全エネルギー収束判定条件。ここで指定する値より全エネルギーの収束が悪いときは、その前のソルバーを続けて使う。デフォルト値は $1e-3/\text{natm hartree}$; ここで <code>natm</code> は原子数
	<code>subspace_rotation</code>		<code>subspace</code> 対角化に関する制御
		<code>subspace_matrix_size</code>	デフォルトの入力値はバンドの数 (<code>num_bands</code>) <code>num_bands</code> よりも大きな値が入力された場合には、強制的に <code>num_bands</code> の値を入力値に設定。
		<code>damping_factor</code>	非対角要素のダンピング係数。[0.0, 1.0] の範囲外の値が入力された場合には、入力値を強制的に 1.0 に設定。
		<code>period</code>	<code>solver</code> タグの <code>submat</code> が ON になっている場合、 <code>period</code> に 1 回 <code>subspace_rotation</code> を行います。例えば <code>period=3</code> のとき <code>iteration(i)</code> のうち、 <code>i=1,4,7,10,...</code> が <code>subspace rotation</code> を行う対象になります。デフォルト値は 1。
		<code>critical_ratio</code>	非対角項の要素の値(1 要素あたり)と対角項の要素の値(1 要素あたり)の比がいったん <code>critical_ratio</code> より小さくなった点に対しては、それ以後 <code>subspace rotation</code> を行いません。デフォルト値は 10^{-15} 。
<code>charge_mixing</code>			電荷混合法。(詳しくは 3.7 と 3.8 を参照)
	<code>mixing_methods</code>		電荷密度の混合法。
		<code>method</code>	選択肢: { <code>simple</code> , <code>broyden2</code> , <code>pulay</code> } デフォルトは <code>pulay</code>
		<code>rmxs</code>	計算開始時の電荷密度を混ぜる割合 デフォルト値は 0.4
		<code>rmxe</code>	<code>itr</code> 回の更新の後に電荷密度を混ぜる割合。 デフォルト値は 0.4。 <code>rmxs</code> の値のみが入力された場合には、 <code>rmxe</code> にも同じ値を適用。
		<code>itr</code>	電荷密度の混合比(<code>rmx</code>) を変化させる回数
		<code>var</code>	<code>rmx</code> を変化させる方法。 選択肢: { <code>linear</code> , <code>tanh</code> }
		<code>prec</code>	前処理の有無。 選択肢: { <code>on</code> , <code>off</code> }
		<code>istr</code>	<code>method</code> が <code>simple</code> 以外の場合に、 <code>istr</code> 回の更新後に、 指定した方法で電荷を混ぜる
		<code>nbmix</code>	蓄えておくべき電荷密度データの回数を指定
		<code>update</code>	<code>nbmix</code> 回分用意されている電荷密度の配列を使い切った時の処理の選択肢。 選択肢: { <code>anew</code> , <code>renew</code> } <code>anew</code> はそれまでのデータを全て棄却して新規に開始。 <code>renew</code> は最も古いデータを最新のデータと入れ換える。
	<code>charge_preconditioning</code>		

		amix	前処理変数 a
		bmix	前処理変数 b
structure_evolution			構造緩和計算用ブロック識別子
		method	選択肢: {sd, quench, gdiis, bfgs, cg, cg2, velocity_verlet, temperature_control}
		dt	時間刻み幅
	stress		ストレス計算
		sw_stress	ストレス計算の有無。選択肢: {on,off}
	gdiis		(GDIIS および BFGS を選択する場合のタグ)
		initial_method	GDIIS (BFGS) へ移行する前に利用する最適化アルゴリズム。選択肢: {quench, cg, sd} デフォルト値は cg
		gdiis_box_size	ここで指定するイオン座標更新回数分のデータを gdiis(bfgs) 用配列に蓄える
		gdiis_hownew	gdiis_box_size で指定した回数分のイオン座標のデータ配列を使い切った時の処理法の選択 選択肢: {anew, renew}
		c_forc2gdiis	GDIIS (BFGS) への切替えの判定条件 デフォルト値は 0.05 (hartree/bohr)
postprocessing			
	dos		状態密度の出力
		sw_dos	状態密度出力の有無。選択肢: {on,off}
		method	選択肢: {tetrahedral, Gaussian}
		deltaE_dos	状態密度出力のエネルギー精度
		variance	method が Gaussian の場合のガウス関数の分散
		nwd_dos_window_width	出力時のエネルギー幅 ΔE を次式で指定: $\Delta E = \text{nwd_window_width} \times \text{deltaE_dos}$
	charge		電荷の出力
		sw_charge_rspace	電荷出力の有無。選択肢: {on,off}
		filetype	電荷出力ファイルの形式 選択肢: {cube, density_only}
		title	電荷の出力ファイルの見出し filetype = cube の時のみ有効
printoutlevel			標準出力への出力レベルの制御 0: 出力なし 1: 情報を出力 2: デバッグ用の情報を出力
		base	他の変数に入力値が指定されていない時は、この値がデフォルト
		pulay	Pulay 電荷混合法
		timing	時間指定情報
		solver	電子状態解法
		evdff	固有エネルギー差
		rmm	残差最小化法
		snl	非局所ポテンシャル
		gdiis	GDIIS 法
		eigenvalue	固有値
		spg	空間群
		kp	k 点
		matdiagon	行列対角法

		vlhxcq	ローカルポテンシャル
		totalcharge	電子密度
		submat	部分空間回転法
		strcfctr	構造因子
		parallel	並列化のための前処理の結果の出力制御
		input_file	入力ファイル F_INP の解析結果の出力
		parallel_debug	1 に設定するとゼロ番ノード以外のプロセスからも output00x_xxx といったファイルに出力を行う。
		jobstatus	計算の進行状況を jobstatus00x に出力
	jobstatus_option		状況ファイルの出力制御
		jobstatus_format	tag, tag_line, table が選択可能。既定値は tag です。
		jobstatus_series	ON または OFF

3.3 全体的な計算条件設定 (Control)

計算をはじめてから実行するのか継続計算を実施するのか、最大どれくらいの時間計算を継続するのか、など、計算全体に関わる条件の設定を **control** ブロックで行います。たとえば、以下のように記述します。

```
control{
  condition = initial
  cpumax = 1 day
  max_iteration = 1000000
}
```

control ブロックにおいては、次の変数を指定することができます。

condition	初期計算か継続計算かなどの指定です。"initial" とすると計算は始めから行われ、"continuation" とすると、波動関数、電荷密度分布などの計算結果を引き継いだ継続計算が行われます。また、"automatic" とすると継続計算に必要な複数のファイル（これらは、前のジョブが正常終了した場合には自動的に生成される）が存在する場合は "continuation"、存在しない場合は "initial" と設定したのと同じ動作をします。"preparation" を指定すると、前処理（使用配列の大きさの評価、k 点生成など）のみ行います。デフォルト値は "automatic" です。 ほかに、(収束した) 電荷密度分布を読み込んで、それを固定したまま波動関数のみを収束させる（バンド分散を計算する場合など）には、"fixed_charge", "fixed_charge_continuation", "fixed_charge_automatic" のいずれかを設定します。それぞれ、最初から計算するか、継続計算するか、自動判定するかの指定です。これらの "initial", "continuation", "automatic", "preparation", "fixed_charge", "fixed_charge_continuation", "fixed_charge_automatic" は、それぞれ、整数 0, 1, -1, -2, 2, 3, -3 で代用することができます。
cpumax	PHASE 計算を実行する時間を、実数+単位、の組合せで指定します。ここで指定した時間を超えると、収束に達していなくても、継続計算用のファイルなどが出力され、計算が停止します。デフォルト値は 86400 s (1 日) です。単位は必須です。使える単位は、"sec", "s" (これは "sec" と同じ), "min", "hour", および "day" です。継続計算になる可能性がある場合には、ジョブの指定時間よりも小さな値に設定しておくのがよいでしょう（例えば、ジョブの制限値が 6 時間で、収束後の状態密度計算などの後処理がなければ、5.8 hour 程度に設定する）。
max_iteration	SCF 計算の総イタレーション数の最大値を指定します。SCF 計算の総イタレーション数がここで指定した数に達すると継続計算用のファイルなどが出力され、計算が停止します。デフォルト値は 10000 です。
max_total_scf_iteration	
max_scf_iteration	構造緩和計算や分子動力学計算における各 MD ステップ内での電子状態の更新回数 (SCF イタレーション数) の最大値を指定します。例えば、構造緩和計算の最初期に構造が不安定で、電子状態に対する収束判定条件を満たすまで数百回に及ぶような大きな回数の SCF イタレーションが必要になることがあります。その場合には、SCF 計算を途中で打ち切って、力を計算してより安定な原子構造に更新してから、次の SCF 計算をすすめた方が有利になります。しかし、あまり小さな値 (10 程度) に設定すると、計算される力の誤差が大きくなり、逆に収束を難しくすることがあるので注意が必要です。正確な力の計算が重要な場合には使用しないでください。

3.4 計算精度の指定 (Accuracy)

3.4.1 カットオフエネルギー

カットオフエネルギーは平面波基底を利用した計算においては計算の信頼性を決める重要なパラメーターです。カットオフエネルギーは以下のように指定します。

```
accuracy{
  cutoff_wf = 25 Rydberg
}
```

```
cutoff_cd = 225 Rydberg
}
```

cutoff_wf 波動関数のカットオフエネルギーをエネルギーの単位で指定します。
cutoff_cd 電荷密度のカットオフエネルギーをエネルギーの単位で指定します。

カットオフエネルギーは十分な精度が得られる値を事前に勘案することが理想的ですが、以下のような指針も有用です。

- **cutoff_wf** はおおよそ **25 rydberg**
- **cutoff_cd** は、ノルム保存型の擬ポテンシャルを利用している場合は **cutoff_wf** の 4 倍、そうでない場合は 9 倍

3.4.2 バンド数

バンド数は、以下のように **accuracy** ブロックの下に **num_bands** 変数によって指定します。

```
accuracy{
  num_bands = 12
}
```

num_bands バンド数

バンド数は、最低限価電子数の半分+1 は必要です。通常最低必要な数の 2 割程度多めの数を採用します。設定値が価電子数の半分に達しない場合には、自動的に設定が増やされます。またこの値を設定していない場合には自動的にバンド数が設定されます。

3.4.3 k 点サンプリングとスメアリング

カットオフエネルギーと同様に、**k** 点サンプリングも計算の信頼性を決める重要なパラメーターです。**k** 点サンプリングは、**accuracy** ブロックの下に **ksampling** ブロックを作成し、**ksampling** ブロックの下で設定を行います。たとえば下記ようになります。

```
accuracy{
  ksampling{
    method = monk
    mesh{
      nx=4
      ny=4
      nz=4
    }
  }
}
```

ksampling ブロックでは、下記の変数/ブロックを定義することができます。

method **k** 点サンプリングの方法を選びます。**monk**, **mesh**, **file**, **gamma**, **directin** のいずれかです。**monk** は Monkhorst-Pack 法によるサンプリングで、通常推奨される方法であり、デフォルト値です。**mesh** は単純なメッシュで逆空間を分割します。四面体法により電荷密度を構成する場合や状態密度の計算を行う場合にはこれを指定します。**file** はファイルから読み込みます。バンド分散をみるために対称線に沿って多くの **k** 点を入力する必要がある場合やフェルミ面の計算において大量の **k** 点を考慮する必要がある場合などに利用します。**gamma** を指定すると Γ 点のみをサンプリングします。充分大きな単位胞を使っていて、 Γ 点のみでも十分な精度が得られる場合には、これを指定します。**directin** は直接 **k** 点の組（個数と座標）を指定します。いずれの方法でも、サンプリング **k** 点に Γ 点が含まれていて、系に反転対称中心がなければ（設定されていない場合）、 Γ 点の波動関数に関する計算は、この点の対称性を利用して他の **k** 点のものに比べて 3 倍程度高速に実行されます（後

mesh 述のとおり、これを抑制する、つまり他の k 点と同じ演算法を適用する手段もあります。
逆空間の分割数を指定します。以下の変数が利用できます。
nx 1 番目の逆格子ベクトルの分割数を指定します。
ny 2 番目の逆格子ベクトルの分割数を指定します。
nz 3 番目の逆格子ベクトルの分割数を指定します。

スメアリングは、フェルミ準位付近の状態を“ぼやかす”操作です。これによって、フェルミ準位付近で状態を持つ金属系においても少ない k 点数で高い精度で計算ができるようになる場合があります。スメアリングは、以下のように **accuracy** ブロックの下の **smearing** ブロックにおいて指定します。

```
accuracy{
  smearing{
    method = parabolic
    width = 0.001 hartree
  }
}
```

smearing ブロックでは以下の変数を利用することができます。

method スメアリングの方法を指定します。parabolic, tetrahedron, cold, improved_tetrahedron のいずれかを指定します。通常利用するのは parabolic で、ガウス関数によってフェルミ準位付近をぼやかします。tetrahedron と improved_tetrahedron は四面体法で、主に四面体法による状態密度計算を行う場合に利用します。cold は Cold スメアリングで、金属において有効とされている方法です。
width スメアリングの幅をエネルギーの単位で指定します。デフォルト値は 0.001 hartree です。このタグは parabolic のときに有効です。tetrahedron の場合、“縮退していると見なされる準位間のエネルギー差の閾値”になります。

3.4.4 交換相関エネルギー

交換相関エネルギーは、LDA と GGA の 2 種類があります。LDA は LDAPW91, PZ、GGA は GGAPBE, RPBE, REVPBE が利用できます。

```
accuracy{
  xctype = ggapbe
}
```

xctype 交換相関エネルギー(LDA, GGA)
LDA : LDAPW91, PZ
GGA : GGAPBE, RPBE, REVPBE

3.4.5 収束判定

収束判定は、電子状態計算の収束判定と構造最適化の際の原子に働く力の収束判定の 2 種類があります。以下のように指定します。

```
accuracy{
  scf_convergence{
    delta_total_energy = 1.0E-8 Hartree
    succession = 3
  }
  force_convergence{
    max_force = 2.0E-4 Hartree/Bohr
  }
}
```

収束判定に関わるブロック/変数は以下の通りです。

scf_convergence	SCF 計算の収束判定を指定するブロックです。
delta_total_energy	全エネルギーの差の閾値をエネルギーの単位で指定します。現在の全エネルギーと 1 ステップ前のエネルギーの差がここで指定した値よりも小さい場合収束判定を満たしたとみなされます。デフォルト値は $1e-9$ です。
succession	delta_total_energy を何回連続でみたせば最終的に収束したと見なすかを指定します。ここで指定した値の回数連続で収束判定を満たせば収束が得られたと判定されます。デフォルト値は 2 です。
force_convergence	原子に働く力の最大値に関する閾値を設定するブロックです。
max_force	原子に働く力の最大値の閾値を力の単位で指定します。デフォルト値は $1e-3$ です。

3.4.6 初期波動関数と初期電荷密度

初期波動関数と初期電荷密度の設定を適切に行うと、電子状態計算を少ない回数で収束させることができます。初期波動関数および初期電荷密度は、以下のように設定することができます。

```
accuracy{
  initial_wavefunctions = atomic_orbitals
  initial_charge_density = atomic_charge_density
  matrix_diagon{
    cutoff_wf = 5 rydberg
  }
}
```

初期波動関数および初期電荷密度の設定に関わるブロック/変数は下記の通りです。

initial_wavefunctions	初期波動関数の設定方法を指定します。 random_numbers , matrix_diagon , file , atomic_orbitals を選択することができます。 random_numbers は乱数による初期化です。 matrix_diagon は行列対角化によってもとめます。この際、初期波動関数作成時にのみ利用するカットオフエネルギーを採用することもできます。その指定方法は後述の matrix_diagon ブロックにおいて行います。 file は、波動関数ファイルから読み込みます。すでにある程度収束した波動関数データファイルを持っている場合はこのオプションを指定し、読み込ませることができます。 atomic_orbitals は、擬ポテンシャルファイルに記録された原子軌道データをもとに初期化を行います。デフォルト値は random_numbers です。
initial_charge_density	初期電荷密度の設定方法を指定します。 Gauss , file , atomic_charge_density のいずれかを選択することができます。 Gauss は原子を中心とした単純なガウス関数による初期化です。 file はファイルから読み込みます。すでにある程度収束した電荷密度データファイルを持っている場合はこのオプションを選択し、読み込ませることができます。 atomic_charge_density は擬ポテンシャルファイルに記録された原子の電荷密度をもとに初期化を行います。デフォルト値は Gauss です。
matrix_diagon	initial_wavefunctions に matrix_diagon を指定している場合に、その振る舞いを制御するためのブロックです。
cutoff_wf	初期波動関数作成時に利用するカットオフエネルギーの値を指定します。デフォルト値は、通常のカットオフエネルギーの半分です。

3.4.7 実空間法

PHASE は、非局所ポテンシャルの演算を逆空間で実行しますが、これを実空間で行わせることも可能です。この機能を利用するためには、以下のように設定します。

```
accuracy{
  nonlocal_potential{
    sw rspace = on
  }
}
```



```

    r0_factor = 1.9
  }
}

```

実空間法は、文献[1]および[2]の方法で実現されています。逆空間法はその演算量が $O(N^3)$ であるのに対し実空間法は $O(N^2)$ なので、大きな系においては実空間法の方が有利となります。ただし、逆空間法では厳密解が得られるのに対し、実空間法は近似解しか得られない点には注意が必要です。`nonlocal_potential` ブロックでは以下のような設定を施すことが可能です。

<code>sw_rspace</code>	実空間法を利用するかどうかを指定します。デフォルト値は <code>off</code> です。
<code>projector_optimization</code>	実空間法を適用するためにはプロジェクターの最適化を行う必要がありますが、その方法を指定します。このパラメーターに <code>prefitting</code> を指定すると[1]の方法で、 <code>mask_function</code> を指定すると文献[2]の方法でこの最適化が行われます。デフォルト値は <code>mask_function</code> です。
<code>r0_factor</code>	「最適化されたプロジェクター」のおよぶ範囲を、もとのプロジェクターの何倍にするかを指定する実数。デフォルト値は 1.9。

- [1] R. D. King-Smith, M. C. Payne, and J. S. Lin, “Real-space implementation of nonlocal pseudopotentials for first-principles total-energy calculations”, *Physical Review B* **44** 13063 (1991).
- [2] Lin-Wang Wang, “Mask-function real-space implementations of nonlocal pseudopotentials”, *Physical Review B* **64** 201107 (2001).

3.5 原子構造 (Structure)

計算に利用するモデルの指定は、`structure` ブロックの下で行います。たとえば、以下のようになります。

```

structure{
  unit_cell_type = Bravais
  unit_cell{
    #units angstrom
    a_vector = 4.914100000 0.000000000 0.000000000
    b_vector = -2.457050000 4.255735437 0.000000000
    c_vector = 0.000000000 0.000000000 5.406000000
  }
  atom_list{
    coordinate_system = Internal
    atoms{
      #units angstrom
      #tag element rx ry rz
      O 0.413100000054 0.145400000108 0.118930000000
      O 0.854599999943 0.267699999886 0.452263333333
      O 0.732300000003 0.586900000006 0.785596666667
      O 0.267699999946 0.854599999892 0.547736666667
      O 0.145399999997 0.413099999994 0.881070000000
      O 0.586899999939 0.732299999879 0.214403333333
      Si 0.530000000000 0.000000000000 0.333333000000
      Si -0.000000000072 0.529999999857 0.666666333333
      Si 0.469999999954 0.469999999908 0.999999666667
    }
  }
  element_list{
    #tag element atomicnumber mass zeta deviation
    O 8 29164.9435 * *
    Si 14 51196.4212 * *
  }
  symmetry{
    method = automatic
  }
}

```

```

        sw_inversion = off
    }
}

```

3.5.1 ユニットセル

unit_cell_type 単位胞の指定方法を設定しています。prmitive か bravais を指定することができます。デフォルト値は bravais です。後述するように、単位胞を格子定数で指定する場合はこの変数を bravais とする必要があります。また、bravais を指定している場合、symmetry ブロックの下で tspace ブロックにおいて定義できる lattice_system 変数によって格子を変換させることが可能です。lattice_system 変数についてはあとの説明も参照してください。

unit_cell 単位胞を指定するブロックです。セルベクトルを指定する方法と格子定数を指定する方法があります。格子定数によって指定する方法は、unit_cell_type が bravais の場合のみ有効です。

- セルベクトルを指定する方法

この方法を利用する場合、ベクトル型データを利用して以下のように記述します

```

unit_cell{
    #units angstrom
    a_vector = a1 a2 a3
    b_vector = b1 b2 b3
    c_vector = c1 c2 c3
}

```

a_vector, b_vector, c_vector によってそれぞれa軸, b軸, c軸をベクトルで指定します。この指定方法の場合、長さの単位はブロック単位で指定する方法のみ利用できる点に 注意してください。この例では、unit_cell ブロックの先頭に#units angstrom とすることによって長さの単位を Å 単位に変更しています。

- 格子定数によって指定する方法

この方法を利用する場合、以下のように記述します。

```

unit_cell{
    a = a0
    b = b0
    c = c0
    alpha = alpha0
    beta = beta0
    gamma = gamma0
}

```

a, b, c, alpha, beta, gamma という変数を利用することによってそれぞれ格子定数 $a, b, c, \alpha, \beta, \gamma$ を指定します。この方法で指定すると、セルベクトルは計算開始時に以下のような“下三角”形式で定義されるようになります。

```

a_vector = a1 0.0 0.0
b_vector = b1 b2 0.0
c_vector = c1 c2 c3

```

3.5.2 原子座標

atom_list 各原子の座標値の指定などを行うブロックです。以下の変数/ブロックを定義することができます。

coordinate_system 原子座標を、カルテシアン座標によって定義するかフラクショナル座標によって定義するかを指定します。internal とするとフラクショナル座標によって、cartesian とするとカルテシアン座標によって指定します。デフォルト値は internal です。

atoms 実際に各原子の座標値を指定する表形式データを記述します。代表的な属性値は以下の通りです。

element 元素名を指定します。元素名は、後述の'element_list'において定義されています。

	る必要があります。
rx	<i>x</i> 座標を指定します。
ry	<i>y</i> 座標を指定します。
rz	<i>z</i> 座標を指定します。
mobile	構造最適化や分子動力学シミュレーションにおいてこの原子が“可動か否か”を指定する真偽値です。可動にしたい場合 on とします。デフォルト値は off です。
mobilex	構造最適化や分子動力学シミュレーションにおいてこの原子の <i>x, y, z</i> 座標を個別に “可動か否か” を指定する真偽値です。可動にしたい場合 on とします。デフォルト値は, mobile に指定された値です。
mobiley	
mobilez	
weight	“重み”を設定します。この属性値に 2 という値を与えた場合, 原点を中心とした反転対称の位置にコピー原子を配置します。デフォルト値は 1 です。2 を与える場合, 系に反転対称性があり, 後述の sw_inversion パラメーターが on となっている必要があります。

3.5.3 原子種の指定

element_list	元素情報を指定するための表形式データを記述するブロックです。代表的な属性値は以下の通り。
element	元素名を指定します。指定は必須です。
atomicnumber	原子番号を指定します。指定は必須です。
mass	質量を指定します。
zeta	スピンを考慮している場合の, 初期スピン分極の値を設定します。
qex	電子を追加/削減する場合に指定します。

擬ポテンシャルファイルは, **file_names.data** ファイルにおいて, ファイルポインター**F_POT(n)** によって指定します。ここで **n** は入力における元素指定の順序に対応する整数です。たとえば, 以下の要領で **O** と **Si** の元素指定が入力ファイルにおいて成されていて,

```
structure{
  ...
  ...
  element_list{
    #tag element atomicnumber mass zeta deviation
    O 8 29164.9435 * *
    Si 14 51196.4212 * *
  }
}
```

対応する擬ポテンシャルファイルが **O** が **O_ggapbe_us_01.pp**, **Si** が **Si_ggapbe_nc_01.pp**, だった場合, **file_names.data** を以下のように記述します。

```
&fnames
F_INP='./nfinp.data'
F_POT(1)='./O_ggapbe_us_01.pp'
F_POT(2)='./Si_ggapbe_nc_01.pp'
/
```

擬ポテンシャルファイルの指定は, 交換相関ポテンシャルの計算方法の指定にも対応しています。公開している擬ポテンシャルファイルの交換相関ポテンシャルの計算方法は **ggapbe** か **ldapw91** のいずれかですが, どちらなのかはファイル名から判定することができます。すなわち, **ggapbe** あるいは **ldapw91** という文字列が擬ポテンシャルファイルのファイル名に含まれています。なお, **ggapbe** と **ldapw91** を混在させた計算を行うことはできませんのでご注意ください。また, 利用できる原子種は 16 種類までです。

電子を追加/削除したい場合の設定もここで行います。qex パラメータに追加したい/削除したい電子数を指定します。たとえば、以下のように指定します。

```
structure{
  ...
  ...
  element_list{
    #tag element atomicnumber mass zeta deviation qex
      O  8 29164.9435 * * +1
      Si 14 51196.4212 * *  0
  }
}
```

追加しているのは電子なので、1 つ追加すると電荷としては 1 減る点に注意が必要です。指定は原子種に対して行いますが、実際は系全体におよぶ設定です。SCF 計算がすすむに従って各原子に分配される余剰電荷は変化します。具体的には波動関数から電荷密度をつくるときに余剰電子 1 個分を含むようにフェルミレベルを調整します。また、Ewald エネルギーはこの余剰電荷を打ち消すように背景に一樣電荷分布をおいた状態にして計算します。

3.5.4 系全体への電荷の付加

系に付加する電荷については、qex 以外にも、additional_charge による指定も可能です。additional_charge で指定する数字が、正の場合は正電荷、負の場合は負電荷（電子）を付け加えます。なお、Ewald エネルギーはこの余剰電荷を打ち消すように背景に一樣電荷分布をおいた状態にして計算します。

下記の例では、系に -1.0 の電荷、すなわち 1 個の電子を付け加えます。

例：additional_charge の使用例

```
structure{
  charged_state{
    additional_charge = -1.0d0
  }
}
```

3.5.5 磁気モーメントの初期値の指定

原子種の初期磁気モーメント（分極）については、3.5.3. 原子種の指定での zeta 以外にも、moment による指定も可能です。単位はボーア磁子 [μ_B] です。なお、moment による指定を行うと、プログラム内部で自動的に zeta 値に変換されます。

下記の例では、Cr1 及び Cr2 の初期磁気モーメントに、それぞれ $3\mu_B$ 及び $-3\mu_B$ が設定されます。

例：moment による初期磁気モーメントの指定

```
structure{
  element_list {
    #tag element atomicnumber moment
      Cr1      24      3.0
      Cr2      24     -3.0
      O  8
  }
}
```

3.5.6 対称性の指定

symmetry	系の対称性を指定するブロックです。対称性を利用することによって、計算量を大幅に減らすことができる場合があります。以下のブロック/変数を利用することができます。
method	対称性指定の方法を指定します。manual と automatic を選ぶことができます。manual を選択すると、生成元を直接入力することによって対称性を指定することができます。automatic を選択すると、PHASE が指定のモデルから自動的に対称性を検知し、計算に反映させます。デフォルト値は manual です。
sw_inversion	系に反転対称性が存在する場合に、それを活用して計算量を減らすかどうかを指定する真偽値です。on の場合反転対称性を利用します。反転対称性の中心は原点(0,0,0)です。このオプションは反転対称性のある系の計算を行う場合は有効にすることが推奨されますが、反転対称性のない系で有効にすると前処理で検知し、終了するのでご注意ください。
tspace	TSPACE を利用して生成元を直接指定するためのブロックです。以下の設定を行います。
lattice_system	unit_cell_type が bravais の場合に、“格子の型”を指定します。選択肢は facecentered, bodycentered, basecentered, rhombohedral です。この変数を指定すると、指定に応じて格子が変換されます。このそれぞれがどのように格子を変換するかについては表 5 を参照してください。この変数を利用することによって、入力ファイルでは指定のしやすいブラベー格子で単位胞を指定しつつ、実際の計算は負荷の少ない基本格子で実行することが可能となります。lattice_system 変数を利用して格子を変換させる場合、変換されるのは単位胞のみです。したがって、原子配置の定義などは基本格子の場合の定義の仕方を採用してください。たとえば、面心立方格子の計算を行う場合面心位置の原子は指定せず、原点の原子のみ指定するようにしてください。また、 k 点サンプリングは変換後の格子に合わせて設定してください。
generators	生成元を指定するテーブルです。生成元は、3 つまでしか定義できないという制約があります。このテーブルでどのように生成元を指定するかについては 4.2 を参照してください。

3.6 波動関数ソルバー (Wavefunction_Solver)

3.6.1 PHASE における計算フロー

PHASE における計算フローを図 3-1 に示します。

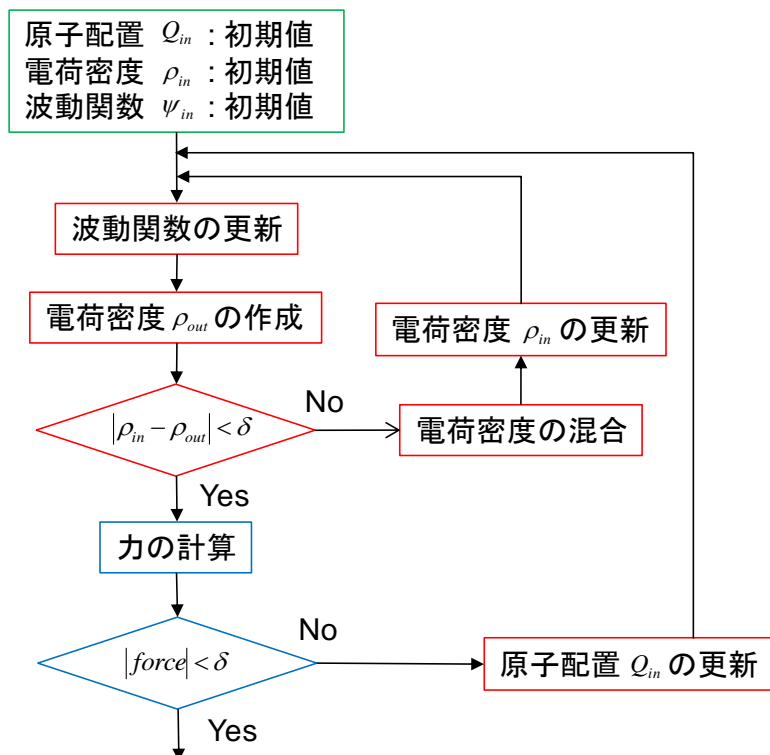


図 3-1 PHASE における計算フロー

波動関数の更新の過程で、Kohn-Sham 方程式

$$(H_{KS}(\rho_{inp}) - \epsilon_i)\psi_i = 0,$$

を解いています。

ある試行の波動関数が与えられ、

$$\Delta_i = (H_{KS}(\rho_{inp}) - \epsilon_i)\psi_i$$

の演算を繰り返し行うことにより、(1)式の解が得られることになります。その際、 Δ_i はエネルギー ϵ の波動関数 ψ に対する勾配と解釈する事ができるので、この勾配が0に近づくように波動関数が更新されます。フローチャート図1中の電荷密度の作成の過程では、更新した波動関数から、新しい電荷密度 ρ が、以下の処方で与えられます。

$$\rho_{out} = 2 \sum_{occ.} |\psi_i|^2,$$

図中の内側のループでは、入力 ρ_{inp} と新しい ρ_{out} が一致するまで計算が行われます。この作業はSCF(自己無動着場)計算と呼ばれています。外側のループでは、与えられた原子配置に対して力の計算が行われ、この力が0(閾値以下)になるような原子配置に到達するまで、計算が続行されます。

3.6.2 波動関数ソルバー

SCF 計算において、「波動関数ソルバー」によって波動関数の更新を繰り返し行います。

波動関数ソルバーは、wavefunction_solver ブロックで指定します。

```
wavefunction_solver{
  solvers{
    #tag sol      till_n prec cmix submat
    pdavidson    2      on 1  on
```

```

        rmm3      -1      on      1      on
    }
    rmm{
        edelta_change_to_rmm = 1e-3
    }
}

```

wavefunction_solver ブロックで利用できるブロック/変数は以下の通りです。

solvers	どの波動関数ソルバーを、どのタイミングで適用するかを設定する重要なテーブルです。以下の属性値を利用することができます。
sol	利用するソルバーのアルゴリズムを指定します。msd, lm+msd, cg, pkosugi, pdavidson, rmm3 から選択します。msd は、修正再急降下法です。1SCF あたりの計算負荷は最も軽い手法ですが、この手法のみで収束を得るのは困難です。主に初期波動関数に対して利用します。lm+msd は msd 法に一次元探索を備えた手法です。1 回あたりの計算負荷は比較的軽い手法で、msd 法と比較して収束の速い手法です。cg 法は共役勾配法です。計算負荷は lm+msd よりも多いですが、収束性は良い場合が多いです。davidson 法は、1 回あたりの計算負荷は高いですが信頼性の高い手法です。pkosugi, pdavidson は davidson 法を並列化した手法であり、通常は davidson 法よりも推奨されます。rmm3 法は 1 回あたりの計算負荷は Davidson 法よりも軽く、信頼性も Davidson 法に劣らない場合の多い手法です。ただし、ランダムな波動関数に適用すると正しい解へ収束しない場合があるので、rmm3 法を利用する場合は 他のソルバーである程度波動関数を収束させてから移行する設定にする必要があります。
till_n	SCF 計算の何ステップ目まで sol を適用するかを指定します。上の例では、till_n 値は、pdavidson は 2 なので 2 回目まで使用する設定となっています（ただし後述の edelta_change_to_rmm で指定する条件を満たさない限り 2 回目以降も pdavidson が利用され続けます）。負の数を指定すると 収束するまでそのソルバーを使い続けます。したがって、rmm3 は最後まで利用される設定となっています。
prec	前処理の有無を真偽値で指定します。デフォルト値は on で、通常変更する必要はありません。
cmix	採用する電荷密度混合法を指定します。電荷密度混合法については 3 において解説します。
submat	部分空間対角化を行うかどうかを真偽値で指定します。デフォルト値は on で、通常変更する必要はありません。
davidson	davidson 法の詳細な振る舞いを設定したい場合に利用するブロックです。以下の変数を利用することができます。
max_subspace_size	davidson 法で利用する部分空間の最大サイズを指定します。デフォルト値はバンド数の 4 倍です。
ndavid	davidson 法はすこしずつ部分空間を拡大しながら波動関数を更新しますが、その回数を指定する変数です。デフォルト値は 5 です。
rmm	rmm 法の詳細な振る舞いを設定したい場合に利用するブロックです。
edelta_change_to_rmm	rmm 法は、ある程度収束した波動関数に適用しないと正しく動作しない場合があります。そこで、ここで指定した値よりも全エネルギーがよかった時点で rmm 法へ移行します。デフォルト値は、1e-3/natm hartree ; ここで natm は原子数。
line_minimization	lm+msd 法や cg 法は 1 次元探索を行い、最適なきざみ幅をもとめます。その 1 次元探索の詳細設定を行うブロックです。
dt_lower_critical	1 次元探索の下限の刻み幅を指定します。デフォルト値は 0.1 です。

dt_upper_critical

1 次元探索の上限の刻み幅を指定します。デフォルト値は 2.0 です。

3.7 電荷密度混合法 (Charge_Mixing)

3.7.1 電荷密度混合法

SCF 計算において、前回の SCF ステップで得られた電荷密度を一定程度混合することによって計算を進行させます。ここでは、この“電荷密度の混合方法”について説明します。電荷密度混合法の設定は、下記のように `charge mixing` ブロックで指定します。

```
charge_mixing{
  mixing_methods{
    #tag method rmxs rmxe prec istr nbmix
    pulay 0.4 0.4 on 3 15
  }
  charge_preconditioning{
    amix = 0.9
    bmix = -1
  }
}
```

`charge mixing` ブロックにおいては、以下のブロック/変数を利用して電荷密度混合法を設定します。

`mixing_methods`

電荷密度混合法を指定するためのテーブルです。電荷密度混合法はいくつでも定義することができます。実際に利用する混合法は、前節の `solvers` テーブルの `cmix` 属性値によって指定します。`cmix` 属性値では、利用したい電荷密度混合法を 1 始まりの定義順で指定します。このテーブルは、以下の属性値をもちます。

`method`

電荷密度混合のアルゴリズムを選びます。`simple`, `broyden2`, `pulay` のいずれかを選択することができます。`simple` は単純混合法です。`broyden2` は Broyden の 2 番目の方法、`pulay` は Pulay による RMM-DIIS 法です。`broyden2` 法と `pulay` 法は、いずれも準ニュートン法の一種です。

`rmxs`

“混合比”の初期値を指定します。

`rmxe`

“混合比”の最終値を指定します。

`prec`

前処理の有無を真偽値で指定します。通常 `on` にします。

`istr`

`broyden2` および `pulay` を選択している場合でも、最初の数ステップは `simple` 法を利用します。この属性値で、この `simple` 法を利用するステップ数を指定します。

`nbmix`

`broyden2` および `pulay` を選択している場合、過去の電荷密度の履歴を利用します。その履歴の大きさを指定します。

`charge_preconditioning`

前処理の係数を設定します。前処理が有効の場合には、次の式を使って、 G の成分毎の混合比を変えます。

$$\rho_{new}(G) \leftarrow (1 - f(G))\rho_{old}(G) + f(G)\rho_{new}(G),$$

$$f(G) = \frac{rmx * amix}{1 + \left(\frac{G_0^2}{G^2}\right)}$$

$$G_0 = bmix * G_{min}$$

ここで G_{min} は、(Γ 点を除いた) G の最小値を指します。この式の `amix` および `bmix` というパラメータを、同名の変数によってこのブロックの下で設定することができます。ただし、通常デフォルト値が推奨されます。

`amix`

`bmix`

3.7.2 収束を加速させるテクニック

ここでは、SCF 計算がなかなか収束しない場合について試すことのできるテクニックを紹介します。

(1) 部分空間対角化

部分空間対角化の有効/無効の設定は、変数 **submat** によって制御することができます。

```
wavefunction_solver{
  solvers{
    #tag      sol      till_n  dts  dte  itr  var      prec  cmix  submat
          lmMSD      -1      0.2  1.0  40   linear  on    1    on
  }
}
```

部分空間対角化の適用を、波動関数を更新する前に適用するか後に適用するかによって収束の振る舞い変化します。これは、特に **RMM** 法を利用している場合に大きな影響を与えます。デフォルトの振る舞いでは波動関数更新前に部分空間対角化が適用されますが、波動関数更新後にする場合には以下のように変数 **before_renewal** を **off** とします（経験的には、**before_renewal=on** の方が収束性がよい場合が多いです）。

```
wavefunction_solver{
  solvers{
    #tag      sol      till_n  dts  dte  itr  var      prec  cmix  submat
          lmMSD      -1      0.2  1.0  40   linear  on    1    on
  }
  submat{
    before_renewal=off
  }
}
```

また、部分空間対角化はバンド数が多い方がより有効に作用します。バンド数を増やせばそれだけ計算量も増えますが、この効果によって全体の計算時間は短くなる場合もあります。

(2) SCF 計算をある回数で打ち切る方法

初期の原子配置が安定な原子配置から遠い場合、SCF 計算を収束させるのに多くの繰り返し計算が必要となる場合があります。このような場合は、たとえ電子状態が十分に収束していなくとも構造最適化をすすめることによって結果的に正しい解へより少ない計算時間で到達することができる場合があります。そこで、入力指定の収束条件を満たしていなくとも収束したとみなし、構造最適化を進める機能が **PHASE** には備わっています。この機能を利用するためには、**control** ブロックの下で **max_scf_iteration** 変数を設定します。

```
control{
  ...
  max_scf_iteration = 50
}
```

この例では、50 回の SCF 計算を行っても収束判定を満たせなかった場合、その時点で至っている電子状態を利用して原子間力を計算し、構造最適化を進行させます。

(3) 電荷密度の差の混合比を変更する方法

スピンを考慮している場合、電荷密度混合は全電荷とスピン電荷密度（アップスピンの電荷密度とダウンスピンの電荷密度の差）に分離して混合します。全電荷とスピン電荷の混合比をそれぞれ違う値に設定することが可能です。このような設定を行うには、下記の要領で **spin_density_mixfactor** 変数を定義します。

```
charge_mixing{
```

```

spin_density_mixfactor = 4
mixing_methods{
#tag      no  method      rmxs  rmxe  prec  istr  nbmix  update
          1   broyden2    0.1 0.1 on 3 15 renew
}
}

```

この例の場合、**spin density mixfactor** は4 であり、電荷密度の差の混合比は $0.1 \times 4 = 0.4$ という値が採用されます。全電荷とスピン電荷を混合するのではなくアップスピンの電荷密度とダウンスピンの電荷密度を直接混合する場合、以下の要領で **sw recomposing** 変数に **off** を設定します。

```

charge_mixing{
    sw_recomposing = off
    ...
}

```

(4) スピン電荷密度の混合に利用するアルゴリズムを変更する
スピン電荷密度に対して、強制的に単純混合法を採用することも可能です。このような設定は、以下のように **spin density** ブロックを作成し、**sw force simple mixing** 変数を定義しその値を **on** とします。

```

charge_mixing{
    sw_recomposing=on
    spin_density_mixfactor = 4
    mixing_methods{
#tag      no  method      rmxs  rmxe  prec  istr  nbmix  update
          1   broyden2    0.1 0.1 on 3 15 renew
    }
    spin_density{
        sw_force_simple_mixing = on
    }
}

```

(5) スピンを固定する方法
一定の間スピンを固定して **SCF** 計算を行うと収束性が改善する場合があります。この設定は、下記の要領で **structure** ブロックの下に **ferromagnetic_state** ブロックを作成し行います

```

structure{
    ...
    ferromagnetic_state{
        sw_fix_total_spin = on
        spin_fix_period = INITIALLY
        total_spin = 1.0
    }
    ...
}

```

ferromagnetic_state ブロックでは以下の変数を利用することができます。

sw_fix_total_spin	“on”とした場合、スピンを固定した計算を行います。
spin_fix_period	スピン固定の方法を指定します。“INITIALLY”と指定した場合、SCF 計算の初期は固定し、すこしずつ拘束を外していきます。“WHOLE”と指定した場合、計算終了までスピンを固定します。整数を指定した場合、その回数だけ固定しあとは通常の計算を行います。
total_spin	アップスピンとダウンスピンの差を指定します。単位胞全体の値を指定してください。

(6) 欠損電荷を混合する方法
PAW 法を利用している場合、欠損電荷の混合が行われます。DFT+U 法を利用している場合、占有行列の混合が行われますが、これも実質上は欠損電荷の混合をおこなっていることと同等です。この混合に対して通常の電荷密度と同様のアルゴリズムで混合させるには、以下のように **charge_mixing** ブロックに

sw_mix_charge_hardpart 変数を定義し、その値を on にします。

```
charge_density{
  ...
  sw_mix_charge_hardpart = on
  ...
}
```

このように設定することによって、PAW 法および DFT+U 法利用時の収束性が向上する場合があります。PAW 法と DFT+U 法を利用する場合は、このパラメータのデフォルト値は on です。

3.8 波動関数ソルバーおよび電荷密度混合法の自動設定

PHASE に搭載されている波動関数ソルバーには、MSD 法、lm+MSD 法、Davidson 法、CG 法、RMM 法、直接対角化法などの基本ソルバーと補助ソルバーとしての subspace rotation があります。さらに、電荷密度混合法として単純混合法、Pulay 法、Broyden による 2 番目の方法などを搭載しています。これらを、問題に応じて適切に組み合わせることによって高速な収束が期待できます。しかし、このように問題に応じて適切に組み合わせるのは非常に手間がかかる作業です。そこで、PHASE には、適切な波動関数ソルバーや電荷密度混合法をプログラムが自動的に選択する機能があります。この機能は、様々な系に対し収束させることができるようになっていますが、もしなかなか収束させられない場合は、手動で波動関数ソルバーや電荷密度ミキサーの設定を行ってください。

「ソルバーセット」は、利用したい計算機能や並列数、バンド数などに応じて自動的に適切なものが採用される仕組みになっているので、利用にあたって特に気にする必要はありません。この自動選択機能は、波動関数に関しては wavefunction_solver ブロックの下で solvers ブロックが、電荷密度に関しては charge_mixing ブロックの下で mixing_methods ブロックが存在しない場合に有効となるので、本機能を利用したい場合は上述の設定を削除するかコメントアウトしてください。wavefunction_solver ブロックは存在していても構わないので、ソルバーに関する詳細設定が必要な場合は対応するサブブロックにおいて行います。たとえば、本機能を利用しつつ rmm ソルバーは収束が 10^{-6} hartree よりもよくなったタイミングで利用したい場合は、以下のような記述を行います。

```
wavefunction_solver{
  rmm{
    edelta_change_to_rmm = 1e-6 hartree
  }
}
```

また、本機能の関連機能として、利用したい電荷密度混合法が 1 種類の場合、以下の簡易表記が利用可能となっています。

```
charge_mixing{
  method = pulay
  rmx = 0.2
  istr = 4
  nbxmix = 10
}
```

各変数は、以下のような意味を持ちます。

method	電荷密度混合の手法を選択します。simple, broyden2, pulay のいずれかが有効。simple は単純混合法、broyden2 法は Broyden による 2 番目の方法、pulay 法は Pulay による DIIS 法です。デフォルト値は pulay。
rmx	混合比を指定します。デフォルト値は 0.4 (スピンを考慮していない場合)、0.1 (スピンを考慮している場合)
istr	broyden2 法ないし pulay 法を採用している場合に、はじめ何回を simple 法で混合するかを指定しま

nbxmix

す。デフォルト値は3 (スピンを考慮していない場合), 5 (スピンを考慮している場合)
broyden2 法ないし pulay 法を採用している場合に、電荷密度の履歴を保持しておく回数を指定します。デフォルト値は15 (スピンを考慮していない場合), 5 (スピンを考慮している場合).

3.9 構造最適化 (Structure_evolution)

構造最適化、分子動力学法計算に関するパラメータは、`structure_evolution` ブロックで指定します。

3.9.1 構造最適化

`structure_evolution` ブロックに、構造最適化の設定をします。

```
...
structure_evolution{
  method = quench
  dt = 50
  ...
}
```

- | | |
|---------------|---|
| method | 構造緩和の方法を指定します。構造緩和のオプションとして、 <code>quench</code> (quenched MD 法), <code>cg</code> (CG 法), <code>cg2</code> 法 (改良 CG 法), <code>gdiis</code> (GDIIS 法), <code>bfgs</code> (BFGS 法) のいずれかが選べます。デフォルト値は <code>bfgs</code> です。 |
| dt | 構造緩和を行う際の時間刻みです。大きい方が早く収束へいたりしますが、大きすぎると計算を正しく進行させることができなくなる場合があります。 <code>quench</code> および <code>gdiis</code> の場合に意味を持ちます。デフォルト値は原子単位で 100 です。構造最適化の場合は、このパラメータは数値解法の都合で導入しているものであり、物理的な意味はありません。 |

GDIIS 法あるいは BFGS 法は原子に働く力が大きい場合安定に計算できない場合があるので、力が大きい内は quenched MD 法か CG 法を利用し、ある程度力が小さくなってから GDIIS(BFGS)法に切り替える、という動作をします。GDIIS(BFGS)に切り替える前の最適化手法と切り替えの判定条件は、それぞれ変数 `initial_method` と `c_forc2gdiis` を利用して 次のように設定します。

```
...
structure_evolution{
  method = gdiis
  dt = 50
  gdiis{
    initial_method = cg
    c_forc2gdiis = 0.0025 hartree/bohr
  }
}
```

ブロック名は、GDIIS, BFGS 共通で `gdiis` です。デフォルト値は `initial_method` が `cg`, `c_forc2gdiis` が 0.05 hartree/bohr です。

- | | |
|-----------------------|---|
| gdiis | GDIIS および BFGS を選択する場合のタグ |
| initial_method | GDIIS (BFGS) へ移行する前に利用する最適化アルゴリズム。選択肢: { <code>quench</code> , <code>cg</code> , <code>cg2</code> , <code>sd</code> }, デフォルト値は <code>cg2</code> |
| gdiis_box_size | ここで指定するイオン座標更新回数分のデータを <code>gdiis(bfgs)</code> 用配列に蓄える |
| gdiis_hownew | <code>gdiis_box_size</code> で指定した回数分のイオン座標のデータ配列を使い切った時の処理法の選択
選択肢: { <code>anew</code> , <code>renew</code> }, デフォルト値は <code>renew</code> |
| c_forc2gdiis | GDIIS (BFGS) への切替えの判定条件
デフォルト値は 0.05 (hartree/bohr) |

3.9.2 分子動力学法計算

分子動力学法計算に関するパラメータは、`structure_evolution` ブロックで指定します。

```
structure_evolution{
  method = velocity_verlet
  dt = 100
}
```

method	原子座標の更新方法を指定する。分子動力学シミュレーションの場合、 <code>velocity_verlet</code> (エネルギー一定の分子動力学シミュレーション) <code>temperature_control</code> (Nosé-Hoover 熱浴による温度一定の分子動力学シミュレーション) <code>velocity_scaling</code> (速度スケーリングによる温度一定の分子動力学シミュレーション)のいずれか
dt	時間刻みを指定する。 デフォルト値は 100 au (約 2.4 fs)
thermostat	分子動力学の場合、構造最適化の場合と異なり物理的な意味のある量である。 熱浴を定義するブロック。
temp	温度を指定する。
qmass	熱浴の質量を指定する。
tdamp	熱浴、熱浴の“周期”を指定する。ここで指定された周期から逆算して熱浴の質量がもとまる。 <code>qmass</code> と <code>tdamp</code> が混在する場合、 <code>qmass</code> の指定が優先される。

3.9.3 構造更新時の電荷密度、波動関数の予測更新 (収束性の向上)

PHASE には、構造最適化や分子動力学シミュレーションを行っている際に、波動関数や電荷密度を原子配置の変化に合わせて“補外”することによって収束性を向上させる機能が備わっています。補外は、文献[1]で紹介されている方法によって行っています。

この機能を利用するには、`structure_evolution` ブロックに `predictor` ブロックを作成し、そこで本機能に関する設定を行います。

```
structure_evolution{
  predictor{
    sw_charge_predictor = on
    sw_extrapolate_charge = on
    sw_wf_predictor = on
  }
}
```

`predictor` ブロックで定義できる変数です。

predictor	
sw_charge_predictor	電荷の予測を行うかどうかを指定するスイッチ。デフォルト値は on
sw_extrapolate_charge	予測の際に電荷密度の補外を行うかどうかを指定するスイッチ。デフォルト値は on
sw_wf_predictor	波動関数の予測を行うかどうかを指定するスイッチ。デフォルト値は off

また、`printoutlevel` に変数 `ipripredictor` を定義し、その値を 2 以上にすると補外をする際に原子配置の予測の

精度やなどの情報がログファイルに出力されます。

- [1] T. Arias, M. C. Payne and J. D. Joannopoulos, “*Ab initio* molecular-dynamics techniques extended to large-length-scale systems”, Physical Review B **45**, 1538 (1992).

3.10 後処理 (Postprocessing)

3.10.1 状態密度 (DOS)

SCF 計算が収束したのち、状態密度の計算を行うことができます。電荷密度の計算を行うには、`postprocessing` ブロックの下に `dos` ブロックで設定します。

```
postprocessing{
  dos{
    sw_dos = on
    method = gaussian
    deltaE_dos = 1e-4 hartree
  }
}
```

`dos` ブロックでは以下の設定を行うことができます。

<code>sw_dos</code>	状態密度計算を行うかどうかを指定する真偽値です。状態密度の計算を行う場合 <code>on</code> とします。
<code>method</code>	状態密度の計算方法を指定します。 <code>gaussian</code> と <code>tetrahedral</code> のいずれかを選択することができます。 <code>gaussian</code> を選択した場合、エネルギー準位をガウス関数によって幅を持たせた上で計算した状態密度が得られます。 <code>tetrahedral</code> の場合四面体法による高精度な状態密度計算を行うことができます。ただし <code>tetrahedral</code> を利用する場合後述の四面体法が利用できる条件もご参照ください。
<code>deltaE_dos</code>	状態密度計算に利用されるエネルギーの幅をハートリー単位で指定します。デフォルト値は <code>1e-4 hartree</code> です。

状態密度の計算方法として `tetrahedral` を利用する場合、以下の条件が満たされている必要があります。

- `k` 点サンプリング手法として `mesh` 法を採用している

```
accuracy{
  ksampling{
    method = mesh
  }
}
```

- `smearing` の方法として `tetrahedral` 法を採用している

```
accuracy{
  smearing{
    method = tetrahedral
  }
}
```

以上が満たされていないと `gaussian` 法による状態密度計算が行われてしまうのでご注意ください。

3.10.2 電荷密度

SCF 計算中は逆空間で電荷密度を扱いますが、収束した電荷密度を実空間に逆フーリエ変換し、出力させることも可能です。こうすることによって `PHASE-Viewer` などを利用して電荷密度の可視化を行うことが可能です。電荷密度を実空間に出力させるためには、`postprocessing` の下の `charge` ブロックで設定を行います。

```
postprocessing{
  charge{
    sw_charge_rspace = on
    filetype = cube
  }
}
```

`charge` ブロックの下では以下の変数の設定を行います。

sw_charge_rspace	電荷密度を実空間で出力するかどうかを指定する真偽値です。on にすると実空間の電荷密度が出力されます。
filetype	電荷密度データのデータフォーマットを指定します。density_only と cube が選べます。density_only の場合電荷密度のみが出力されます。デフォルト値は density_only です。cube の場合、Gaussian Cube 形式で電荷密度が出力されます。このパラメータは、cube に設定することを推奨します。
title	Gaussian Cube ファイルの“見出し”を指定します。空白文字を含める場合、全体を半角の 2 重引用符で囲みます。

また、filetype として cube を選択した場合、file_names.data ファイルにおいて電荷密度ファイルのファイル名を変更しておくことを推奨します。

```
&fnames
...
F_CHR = './nfchr.cube'
/
```

変更しない場合のデフォルト値は nfchr.data です。

スピン分極を考慮している場合は、file_names.data で指定したファイル名が nfchr.cube であったとすると、nfchr.up.cube と nfchr.down.cube という 2 つのファイルにそれぞれスピナップ・ダウン に対応する電荷密度データが出力されます。

3.10.3 構造最適化/分子動力学シミュレーションの最中に後処理を行う方法

ここで説明した後処理は、特に設定が施されていない場合力が収束した後に実行されます。構造最適化や分子動力学シミュレーションの最中に後処理を行いたい場合、以下のような記述を行います。

```
postprocessing{
...
frequency = 5
}
```

変数 frequency に正の値を指定した場合、指定した回数に 1 回の頻度で後処理が行われるようになります。結果は、状態密度の場合は dos_iterxx.data ファイル、電荷密度は nfchr_iterxx.data ファイルに記録されます (xx は原子配置の更新回数に相当する数値に読み替えてください)。なお、この機能によって出力できるのは、状態密度と電荷密度のみです。

3.11 ログレベル (PrintLevel)

PHASE は output000 というファイル (000 は計算を行うたびに 1 ずつ増えます) にログを記録します。そのログの詳細度の設定は printoutlevel ブロックで行います。

```
printoutlevel{
base = 1
}
```

最上位に printoutlevel ブロックを作成し、その下にログレベルを制御する変数を定義します。ログレベルを制御するための変数は 0,1,2 のいずれかの値を取り、数字が大きいほどより詳細な出力が得られます。デフォルト値はすべて 1 です。ログレベルを制御する変数として主なものは以下の通りです。

base	計算全体のログレベルを指定します。特に指定のない項目はここでの指定に従います。
timing	計算時間に関わるログレベルを制御します。
input	入力に関わるログレベルを制御します。
solver	波動関数ソルバーに関わるログレベルを制御します。

spg 空間群に関わるログレベルを制御します。

base=2 に設定すると膨大な量の出力が得られ、ログファイルが見つらなくなってしまいます。得られる情報のほとんどはデバッグ情報なので、特別な事情がない限り base=2 は指定しないことを推奨します。

4. 基本機能を利用した計算例

4.1 全エネルギー計算

最も基本的な計算機能として、全エネルギーの計算があります。複数の格子定数で全エネルギーを計算し格子定数や体積弾性率を計算することや、絶対零度における結晶の安定性を評価することができます。

4.1.1 入力パラメータ

シリコン結晶（ダイヤモンド構造）の全エネルギー計算を例とします。シリコン原子 8 個の系 Si8 を対象とします。シリコン結晶（ダイヤモンド構造）の構造を図 4-1 に示します。

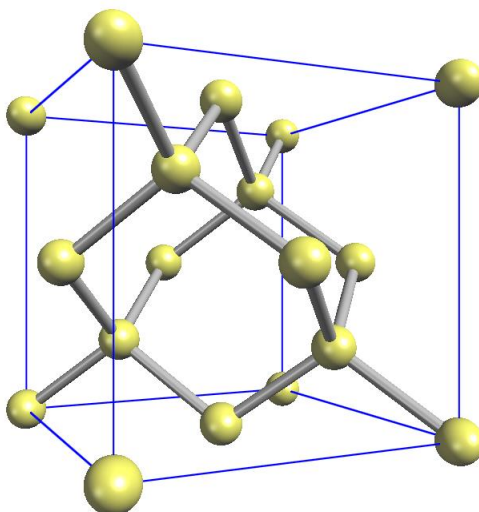


図 4-1 シリコン原子が構成するダイヤモンド構造

計算に使う入力ファイルは、ファイル file_names.data の中で指定します。

file_names.data は以下のように記述します。

```
&fnames
  F_INP      = './input_scf_Si8.data'
  F_POT(1)   = '../pp/Si_ldapw91_nc_01.pp'
  ...
  F_CHR      = './nfchr.cube'
&end
```

PHASE を実行するためには、擬ポテンシャルデータ F_POT(1) と、入力ファイル F_INP が指定されている必要があります。 Si_ldapw91_nc_01.pp はシリコンの擬ポテンシャル・データです。

入力パラメータファイル `input_scf_Si8.data` について説明します。

Control ブロックでは、全体的な計算条件を指定します。`cpumax` は計算時間の最大値を指定しています。

```
Control{
    condition = initial
    cpumax = 3600 sec      ! {sec|min|hour|day}
}
```

Accuracy ブロックでは、計算精度を指定します。

```
accuracy{
    cutoff_wf = 9.00 rydberg
    cutoff_cd = 36.00 rydberg
    num_bands = 20
    ksampling{
        method = mesh ! {mesh|file|directin|gamma}
        mesh{ nx = 4, ny = 4, nz = 4 }
    }
    ...
    xctype = ldapw91
    scf_convergence{
        delta_total_energy = 1.e-12 hartree
        succession = 3
    }
    ...
}
```

`cutoff_wf` と `cutoff_cd` は、波動関数と電荷密度分布のカットオフ・エネルギーが、それぞれ 9.0Ry と 36.0Ry という値であることを表しています。

`num_bands` はエネルギー準位数を表します。この計算では、Si 原子 8 個を扱いますが、各原子は 4 個の価電子をもつため、占有される準位数は、スピンの縮退度を考慮すると $8 \times 4 / 2 = 16$ となります。このため `num_bands` は、17 以上に設定しておく必要があります。また、`ksampling` というタグは、 k 点のサンプリングの方法を指定するのに使われます。この例では、 $4 \times 4 \times 4$ のメッシュ点が k 点サンプリングとなります。

`xctype = ldapw91` では、LDA 型交換相関エネルギーを指定しています。

`scf_convergence` では、計算の収束条件を指定します。この例の場合、全エネルギーの計算誤差が 10^{-12} Hartree 未満に収まるという結果が連続して 3 回続いたら、計算を終了させるように指定されています。

Structure ブロックでは、結晶構造を指定します。単位はデフォルトが原子単位となっています（長さの単位は Bohr）。

```
structure{
    unit_cell_type = primitive
    unit_cell{
        a_vector = 10.26    0.00    0.00
        b_vector = 0.00    10.26    0.00
        c_vector = 0.00    0.00    10.26
    }
    atom_list{
        coordinate_system = internal ! {cartesian|internal}
        atoms{
            #default weight = 1, element = Si, mobile = 1
            #tag    rx      ry      rz
            0.125    0.125    0.125
            -0.125   -0.125   -0.125
            0.125    0.625    0.625
            -0.125   -0.625   -0.625
            0.625    0.125    0.625
            -0.625   -0.125   -0.625
            0.625    0.625    0.125
            -0.625   -0.625   -0.125
        }
    }
}
```

```

    }
    element_list{ #tag element  atomicnumber
                  Si            14
    }
}

```

atom_list では、原子種、単位胞内での内部座標位置、それぞれの原子の位置を固定するか否かを指定します。
element_list では、元素名とその原子番号を指定します。

Postprocessing ブロックでは、後処理のパラメータを指定します。

```

postprocessing{
    ...
    charge{
        sw_charge_rspace      = ON
        filetype               = cube  !{cube|density_only}
        title                  = "This is a title line for the bulk Si"
    }
}

```

charge ブロックでは、電荷密度の出力について指定します。電荷密度は、file_names.data において F_CHR で指定したファイルに出力されます。filetype=cube とする事により、Gaussian cube 形式で出力されます。このとき、F_CHR で指定されるファイル名は、*.cube の形式である必要があります。Gaussian cube ファイルは、PHASE Viewer などの可視化ソフトウェアを使って可視化表示することが可能です。

4.1.2 計算の実行

PHASE を以下のように実行します。

```
% mpirun -np NP ../../bin/phase ne=NE nk=NK
```

ここで、NP, NE, NK はそれぞれ、計算に使用するプロセッサの数、エネルギー準位の分割計算の数、および、 k 点の分割計算の数を指します。これらのパラメータの値の間には、 $NP = NE \times NK$ という関係が成り立っていなければなりません。

また、1 CPU の計算機を使う場合には、以下のように実行します。

```
% mpirun ../../bin/phase
```

計算の途中経過を確認するには、計算のログ出力ファイル output000 に出力されている全エネルギーの収束状況を調べます。以下のように実行すると、全エネルギーに関する部分を抽出できます。

```
% grep TOTAL output000
```

Si8 のサンプルを使って得られた output000 では、次のような結果が表示されます。

TOTAL ENERGY FOR	1 -TH ITER=	-30.829890224786	edel =	-0.308299D+02	: SOLVER =	MATDIAGON
TOTAL ENERGY FOR	2 -TH ITER=	-31.552279425330	edel =	-0.722389D+00	: SOLVER =	DAVIDSON
TOTAL ENERGY FOR	3 -TH ITER=	-31.585338309210	edel =	-0.330589D-01	: SOLVER =	DAVIDSON
TOTAL ENERGY FOR	4 -TH ITER=	-31.587690531430	edel =	-0.235222D-02	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	5 -TH ITER=	-31.587917448876	edel =	-0.226917D-03	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	6 -TH ITER=	-31.587936739174	edel =	-0.192903D-04	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	7 -TH ITER=	-31.587937104439	edel =	-0.365265D-06	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	8 -TH ITER=	-31.587937141798	edel =	-0.373598D-07	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	9 -TH ITER=	-31.587937146347	edel =	-0.454873D-08	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	10 -TH ITER=	-31.587937147067	edel =	-0.720142D-09	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	11 -TH ITER=	-31.587937147180	edel =	-0.112617D-09	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	12 -TH ITER=	-31.587937147235	edel =	-0.548042D-10	: SOLVER =	SUBMAT + RMM3

SCF 計算において、全エネルギーの値が収束してゆく様子が分かります。

4.1.3 計算結果の出力

計算された全エネルギーは、F_ENF ファイルに出力されます。

Si8 の例題では、F_ENF ファイル(ファイル名 : nfefn.data) は以下のようになっています。

iter_ion,	iter_total,	etotal,	forcmx
1	12	-31.5879371472	0.0000003022

計算が終了すると、電荷密度ファイル **nfchr.cube** が作成されます。電荷密度分布を図 4-2 に示します。原子数を増やすなど、**cube file** に修正を加えています。

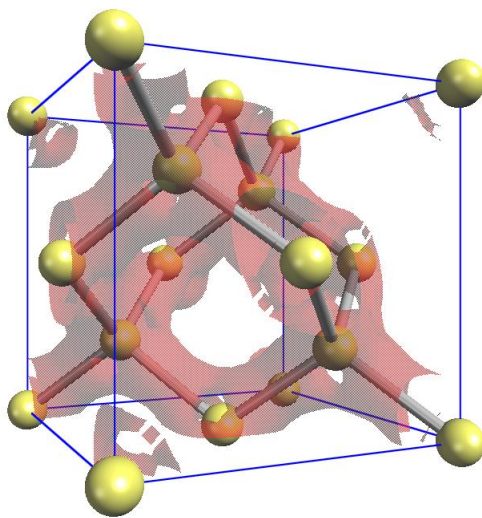


図 4-2 シリコン結晶の電荷密度分布

4.2 対称性を考慮した計算

PHASE には、結晶の対称性を考慮することによって計算量を低減する機能があります。対称性は、自動的に判定させることも可能ですし、生成元を直接指定することによって明示的に指定することも可能です。原子配置を指定する方法には、基本格子を指定する方法とブラベー格子を指定する方法があります。具体的には、変数 `unit_cell_type` の入力値を `primitive` か `Bravais` のどちらかから選択することで指定できます。

4.2.1 入力パラメータ

4.2.1.1 単位胞の指定

(1) 単位胞を基本格子で指定

```
unit_cell_type = primitive
unit_cell{
  #units bohr
  a_vector = 0.00000 5.13000 5.13000
  b_vector = 5.13000 0.00000 5.13000
  c_vector = 5.13000 5.13000 0.00000
}
```

この方法は、`unit_cell_type` が `primitive` でも `Bravais` でも使用できます。

(2) 単位胞を格子定数で指定

```
unit_cell_type = Bravais
unit_cell{
  #units bohr
  a = 10.26, b = 10.26, c = 10.26
  alpha = 90, beta = 90, gamma = 90
}
```

この方法は、`unit_cell_type` が `Bravais` の時のみ使用できます。ブラベー格子を指定して入力した場合、対称性の指定により、プログラム内で基本格子を決定します。計算は、プログラムが決定した基本格子を元に行われるので、原子座標の指定、 k 点分割数や、バンド計算時の対称 k 点の指定などは、この基本格子を元に行う必要がある点に注意ください。

`unit_cell_type` として `Bravais` を利用する場合、副格子点に位置する原子は指定しないようにしてください。たとえば体心原子を含む結晶の場合、 $(0, 0, 0)$ の原子は指定し、 $(0.5, 0.5, 0.5)$ の原子は指定しないようにしてください。`Bravais` を利用する際に指定が必要な結晶の型は、`lattice_system` という変数で指定します。表 4.1 を参照してください。

菱面体晶系(rhombohedral) の場合には、対応する六方晶系(hexagonal) の格子定数を入力します。六方晶系と菱面体晶系の基本並進ベクトルの関係を図 4-3 に示します。

原子座標を内部座標で入力する場合は、等価原子を除いて、単位胞内のすべて原子の位置を結晶軸ベクトル(慣用単位胞の 3 辺を表すベクトル) に対する相対座標(ワイコフ位置の原子座標) で入力します。デカルト座標で入力する場合は、表 4.2 に示されている基本並進ベクトルと整合するように入力してください。

表 4.1 ブラベー格子と晶系

晶系	格子定数	unit_cell に記述する値	格子の種類	lattice_system に指定する単語
立方(c)	a	$a=a, b=a, c=a$	単純(P)	primitive

		$\alpha=90, \beta=90, \gamma=90$	面心(F) 体心(I)	facecentered bodycentered
正方(t)	a, c	$a=a, b=a, c=c$ $\alpha=90, \beta=90, \gamma=90$	単純(P) 体心(I)	primitive bodycentered
直方(o)	a, b, c	$a=a, b=b, c=c$ $\alpha=90, \beta=90, \gamma=90$	単純(P) 底心(C) 面心(F) 体心(I)	primitive basecentered facecentered bodycentered
六方(h)	a, c	$a=a, b=a, c=c$ $\alpha=90, \beta=90, \gamma=120$	単純(P)	hexagonal
三方(h) 菱面体軸 六方晶軸	a, c	$a=a, b=a, c=c$ $\alpha=90, \beta=90, \gamma=120$	菱面(R) 単純(P)	rhombohedral hexagonal
単斜(m)	a, b, c β	$a=a, b=b, c=c$ $\alpha=90, \beta=\beta, \gamma=90$	単純(P) 底心(C)	primitive basecentered
三斜(a)	a, b, c α, β, γ	$a=a, b=b, c=c$ $\alpha=\alpha, \beta=\beta, \gamma=\gamma$	単純(P)	primitive

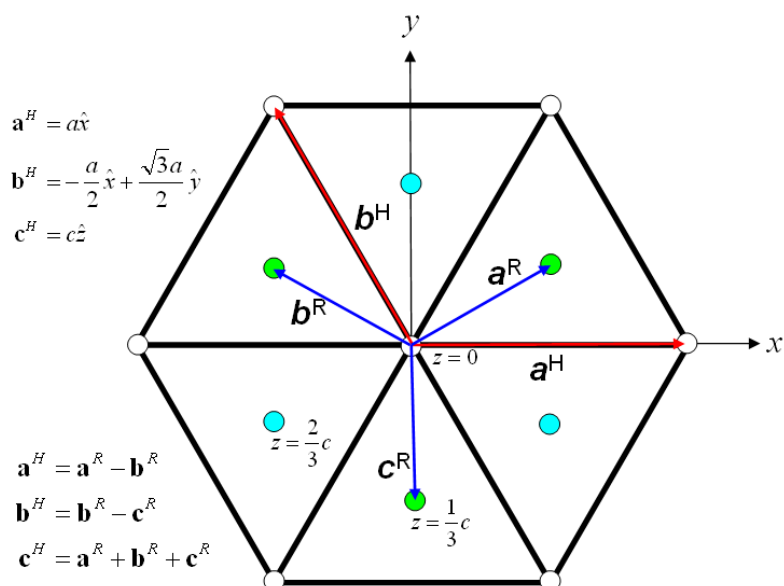


図 4-3 六方晶系と菱面体晶系の関係. 六方軸の方から見た格子点と基本並進ベクトルが示されています.

$\mathbf{a}^H, \mathbf{b}^H, \mathbf{c}^H$ は六方晶系の基本並進ベクトルで, $\mathbf{a}^R, \mathbf{b}^R, \mathbf{c}^R$ は菱面体晶系の基本並進ベクトルです.

表 4.2 ブラベー格子の基本並進ベクトル.

ブラベー格子	\mathbf{a}	\mathbf{b}	\mathbf{c}
単純立方(cP)	$a\hat{x}$	$a\hat{y}$	$a\hat{z}$
面心立方(cF)	$\frac{a}{2}(\hat{y} + \hat{z})$	$\frac{a}{2}(\hat{x} + \hat{z})$	$\frac{a}{2}(\hat{x} + \hat{y})$
体心立方(cI)	$\frac{a}{2}(-\hat{x} + \hat{y} + \hat{z})$	$\frac{a}{2}(\hat{x} - \hat{y} + \hat{z})$	$\frac{a}{2}(\hat{x} + \hat{y} - \hat{z})$
単純正方(tP)	$a\hat{x}$	$a\hat{y}$	$c\hat{z}$
体心正方(tI)	$\frac{1}{2}(-a\hat{x} + a\hat{y} + c\hat{z})$	$\frac{1}{2}(a\hat{x} - a\hat{y} + c\hat{z})$	$\frac{1}{2}(a\hat{x} + a\hat{y} - c\hat{z})$
単純直方(oP)	$a\hat{x}$	$b\hat{y}$	$c\hat{z}$
底心直方(oC)	$\frac{1}{2}(a\hat{x} - b\hat{y})$	$\frac{1}{2}(a\hat{x} + b\hat{y})$	$c\hat{z}$
面心直方(oF)	$\frac{1}{2}(b\hat{y} + c\hat{z})$	$\frac{1}{2}(a\hat{x} + c\hat{z})$	$\frac{1}{2}(a\hat{x} + c\hat{y})$

体心立方(oI)	$\frac{1}{2}(-a\hat{x} + b\hat{y} + c\hat{z})$	$\frac{1}{2}(a\hat{x} - b\hat{y} + c\hat{z})$	$\frac{1}{2}(a\hat{x} + b\hat{y} - c\hat{z})$
単純六方(hP)	$a\hat{x}$	$a(-\frac{1}{2}\hat{x} + \frac{\sqrt{3}}{2}\hat{y})$	$c\hat{z}$
単純菱面体(hR)	$\frac{a}{2}\hat{x} + \frac{a}{2\sqrt{3}}\hat{y} + \frac{1}{3}c\hat{z}$	$-\frac{a}{2}\hat{x} + \frac{a}{2\sqrt{3}}\hat{y} + \frac{1}{3}c\hat{z}$	$-\frac{a}{\sqrt{3}}\hat{y} + \frac{1}{3}c\hat{z}$
単純単斜(mP)	$a\hat{x}$	$b\hat{y}$	$c(\cos \beta \hat{x} + \sin \beta \hat{z})$
底心単斜(mC)	$\frac{1}{2}(a\hat{x} - b\hat{y})$	$\frac{1}{2}(a\hat{x} + b\hat{y})$	$c(\cos \beta \hat{x} + \sin \beta \hat{z})$
単純三斜(aP)	$a\hat{x}$	$b(\cos \gamma \hat{x} + \sin \gamma \hat{y})$	$c\left(\cos \beta \hat{x} + \frac{\cos \alpha - \cos \beta \cos \gamma}{\sin \gamma}\hat{y} + \sqrt{1 - \frac{\cos^2 \alpha + \cos^2 \beta - 2 \cos \alpha \cos \beta \cos \gamma}{\sin^2 \gamma}}\hat{z}\right)$

4.2.1.2 対称性の指定

対称性の指定のやり方には、結晶構造を入力する方法、対称操作を自動的に決定する方法、生成元を入力する方法があります。

(1) 結晶構造を入力する方法

変数 `crystal_structure` に、結晶構造の型を入力します。この場合、選択肢には `diamond`, `hexagonal`, `fcc`, `bcc`, `simple_cubic` の5つがあります。Si 結晶の場合に指定する結晶構造は `diamond` です。

(2) 対称操作を自動的に決定する方法

`method` 変数に `automatic` を指定することで、対称性は自動的に決定されます。`tspace` ブロックの `lattice_system` の指定は、`primitive` の場合以外は指定することが推奨されます。

```

symmetry{
  method = automatic
  tspace{
    lattice_system = facecentered
    !{rhombohedral|hexagonal|primitive|facecentered|bodycentered|basecentered}
  }
}

```

(3) 生成元を入力する方法

生成元は、`tspace` ブロックで指定します。Si 結晶の場合、`tspace` の入力値は以下のようになります。

```

tspace{
  lattice_system = facecentered
  !{rhombohedral|hexagonal|primitive|facecentered|bodycentered|basecentered}
  num_generators = 3
  generators{
    #tag rotation  tx   ty   tz
    IE           0    0    0
    C31+         0    0    0
    C4X+        1/4  1/2  3/4
  }
}

```

予め、面心格子を使うことを `lattice_system = facecentered` で、また、生成元の数3であることを `num_generators = 3` で宣言した後で、タグ `generators` の中で、`IE`, `C31+`, `C4X+` が、具体的に3種類の生成元を指定しています。

生成元の指定の方法を説明します。

生成元の回転操作は、以下のコードで指定します。各行は、それぞれ一つの回転操作に対応します。一列目の数

字か二列目の記号を利用して **generators** テーブルの **rotation** 列に対称操作を指定します. 三列目から五列目までが対応する回転操作を表します. なお, 三方晶, 六方晶の場合に現れている **w** は **x-y** を表します. コードは, 一列目の数字でも二列目の文字列でも指定することが可能です.

三方晶, 六方晶の場合.

1	E	X	Y	Z	13	IE	-X	-Y	-Z
2	C6+	W	X	Z	14	IC6+	-W	-X	-Z
3	C3+	-Y	W	Z	15	IC3+	Y	-W	-Z
4	C2	-X	-Y	Z	16	IC2	X	Y	-Z
5	C3-	-W	-X	Z	17	IC3-	W	X	-Z
6	C6-	Y	-W	Z	18	IC6-	-Y	W	-Z
7	C211	-W	Y	-Z	19	IC211	W	-Y	Z
8	C221	X	W	-Z	20	IC221	-X	-W	Z
9	C231	-Y	-X	-Z	21	IC231	Y	X	Z
10	C212	W	-Y	-Z	22	IC212	-W	Y	Z
11	C222	-X	-W	-Z	23	IC222	X	W	Z
12	C232	Y	X	-Z	24	IC232	-Y	-X	Z

三方晶, 六方晶以外の場合

1	E	X	Y	Z	25	IE	-X	-Y	-Z
2	C2X	X	-Y	-Z	26	IC2X	-X	Y	Z
3	C2Y	-X	Y	-Z	27	IC2Y	X	-Y	Z
4	C2Z	-X	-Y	Z	28	IC2Z	X	Y	-Z
5	C31+	Z	X	Y	29	IC31+	-Z	-X	-Y
6	C32+	-Z	X	-Y	30	IC32+	Z	-X	Y
7	C33+	-Z	-X	Y	31	IC33+	Z	X	-Y
8	C34+	Z	-X	-Y	32	IC34+	-Z	X	Y
9	C31-	Y	Z	X	33	IC31-	-Y	-Z	-X
10	C32-	Y	-Z	-X	34	IC32-	-Y	Z	X
11	C33-	-Y	Z	-X	35	IC33-	Y	-Z	X
12	C34-	-Y	-Z	X	36	IC34-	Y	Z	-X
13	C2A	Y	X	-Z	37	IC2A	-Y	-X	Z
14	C2B	-Y	-X	-Z	38	IC2B	Y	X	Z
15	C2C	Z	-Y	X	39	IC2C	-Z	Y	-X
16	C2D	-X	Z	Y	40	IC2D	X	-Z	-Y
17	C2E	-Z	-Y	-X	41	IC2E	Z	Y	X
18	C2F	-X	-Z	-Y	42	IC2F	X	Z	Y
19	C4X+	X	-Z	Y	43	IC4X+	-X	Z	-Y
20	C4Y+	Z	Y	-X	44	IC4Y+	-Z	-Y	X
21	C4Z+	-Y	X	Z	45	IC4Z+	Y	-X	-Z
22	C4X-	X	Z	-Y	46	IC4X-	-X	-Z	Y
23	C4Y-	-Z	Y	X	47	IC4Y-	Z	-Y	-X
24	C4Z-	Y	-X	Z	48	IC4Z-	-Y	X	-Z

他方, 回転に伴う並進操作は **generators** テーブルの **tx**, **ty**, **tz** 列にそれぞれ指定します. 格子ベクトルを基準に分数で入力してください.

4.2.1.3 反転対称性がある場合

反転対称性がある場合, これを考慮する事により, 計算量を減らすことができます. たとえば, 以下の座標データは原点を中心として反転対称性があるので, それを考慮するように設定すると計算量を減らすことができます.

```
atom_list{
  coordinate_system = internal ! {cartesian|internal}
  atoms{
    #units !{angstrom(cartesian) | bohr(cartesian)}
    #tag  rx      ry      rz      weight  element  mobile
          0.125    0.125    0.125      1        Si        1
  }
}
```

-0.125	-0.125	-0.125	1	Si	1
}					
}					

反転対称性を考慮する設定は、**symmetry** ブロックの下で **sw_inversion = on** とすることによって行います。

```
structure{
  ...
  symmetry{
    ...
    sw_inversion = on
  }
  ...
}
```

また、反転対称性を考慮する場合、原子配置の **weight** 属性値を利用することによって座標データ入力を省力化することも可能です。たとえば、以下の指定は **sw_inversion=on** の場合上記の座標例と等価です。

```
atom_list{
  coordinate_system = internal ! {cartesian|internal}
  atoms{
    #units !{angstrom(cartesian) | bohr(cartesian)}
    #tag rx ry rz weight element mobile
        0.125 0.125 0.125 2 Si 1
  }
}
symmetry{
  sw_inversion = on
}
```

weight 属性値が 2 の原子は、反転対称位置に自分自身のコピーが作成されます。

第 2 節で指定した対称群に反転対称操作が含まれる場合、この **option** を指定することを推奨します。なお、原子座標を入力する場合反転対称操作の中心は、原点であることにご注意ください。また、反転対称性のない系において **sw_inversion = on** を指定するとエラーメッセージを出力して計算を終了します。

4.2.2 計算例：シリコン結晶(Si2)

シリコン原子が構成するダイヤモンド構造の基本格子は原子 2 個を含みます。ここでは、シリコン原子 2 個からなる Si_2 という系を例とします。図 4-4 は Si_2 の原子構造です。
計算例題は、sample/Si2/です。

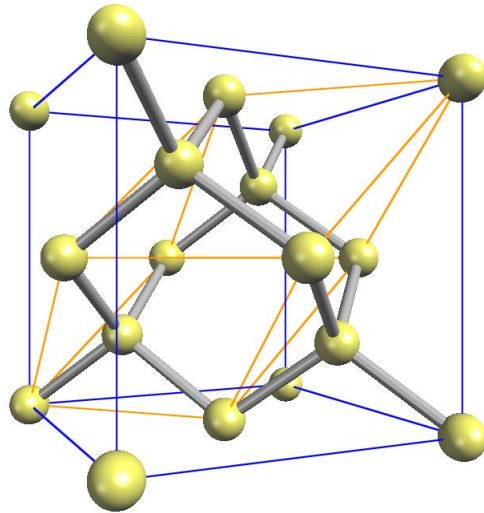


図 4-4 Si_2 の原子構造。黄線は原子 2 個を含む基本格子を表す

(1) SCF 計算

SCF 計算を行い、電荷密度を計算します。計算例題は sample/Si2/scf です。

ファイル file_names.data において、入力パラメータファイルと擬ポテンシャルを指定します。

```
F_INP      = './input_scf_Si.data'
F_POT(1)   = '../..pp/Si_ldapw91_nc_01.pp'
F_CHGT     = '../scf/nfchgt.data'
...
```

入力パラメータファイルにおいて、crystal_structure を diamond として、対称性を指定します。

```
accuracy{
    cutoff_wf = 9.00 rydberg
    cutoff_cd = 36.00 rydberg
    num_bands = 8
}

structure{
    unit_cell_type = Bravais
    unit_cell{
        a = 10.26, b = 10.26, c = 10.26
        alpha = 90, beta = 90, gamma = 90
    }
    symmetry{
        crystal_structure = diamond
    }
}
```

```

atom_list{
  atoms{
    #tag    rx      ry      rz    element
          0.125    0.125    0.125     Si
          -0.125   -0.125   -0.125     Si
  }
}
}

```

エネルギー準位数を表す `num_bands` の値は、原子数が2個なので8としています。

PHASE を実行します。

```
% mpirun ../.../bin/phase
```

計算が終了すると、`file_names.data` というファイルの中で、変数 `F_CHGT` で指定した出力ファイル `nfchgt.data` に、計算によって得られた電荷の情報が出力されます。

(2) 状態密度(DOS)の計算

状態密度 (DOS) を計算します。計算例題は、`sample/Si2/dos` です。

計算結果の出力ファイルが上書きされるのを避けるため、SCF 計算を行ったディレクトリとは別のディレクトリで実行します。

SCF 計算結果の電荷密度ファイル `nfchgt.data` を使います。擬ポテンシャルはSCF計算と同じものを使います。

`file_names.data` では、入出力ファイルを以下のように指定しています。

```

F_INP      = './input_dos_Si.data'
F_POT(1)   = '../.../pp/Si_ldapw91_nc_01.pp'
...
F_CHGT     = '../scf/nfchgt.data'
...
F_ENERG    = './nfenergy.data'
...

```

`F_CHGT` で指定している電荷密度のデータは、SCF 計算で得られた出力ファイルです。

入力ファイルは `input_dos_Si.data` と `nfchgt.data` の2つです。

入力ファイル `input_dos_Si.data` について、SCF 計算の入力ファイル `input_scf_Si.data` と異なる部分を以下に示します。

```

Control{
  condition = fixed_charge
}

accuracy{
  cutoff_wf = 9.00 rydberg
  cutoff_cd = 36.00 rydberg
  num_bands = 8
  ksampling{
    method = mesh
    mesh{ nx = 4, ny = 4, nz = 4 }
  }
  smearing{
    method = tetrahedral
  }
  xctype = ldapw91
  initial_wavefunctions = matrix_diagon
  matrix_diagon{
    cutoff_wf = 9.00 rydberg
  }
  ek_convergence{

```

```

        num_max_iteration = 200
        sw_eval_eig_diff = on
        delta_eigenvalue = 1.e-8 hartree
        succession      = 2
    }
}

postprocessing{
    dos{
        sw_dos      = ON
        method      = tetrahedral    !{ tetrahedral | Gaussian }
        deltaE_dos   = 1.e-3 eV
        nwd_window_width = 10
    }
}

```

最初のタグである **Control** の部分で、SCF 計算で得られた電荷の分布を固定して使用することを指定します。ksampling では k 点サンプリングが $4 \times 4 \times 4$ 、smearing では四面体法を用いること、ek_convergence では収束条件を指定しています。Postprocessing ブロックでは、計算終了後の後処理として、四面体法による状態密度の計算のパラメータが指定されています。

これらの入力ファイルを使って、プログラム **ekcal** を用いて、状態密度の計算を行います。

```
% mpirun ../../../../bin/ekcal
```

計算を実行すると、**nfenergy.data** という出力ファイルが生成されます。これは、各 k 点ごとのエネルギー値を、エネルギーの低い方から順に 出力したもので、その最初の部分は以下のようになっています。

```

num_kpoints =    141
num_bands    =     8
nspin        =     1
Valence band max = 0.233846

=== energy_eigen_values ===
ik =    1 ( 0.000000  0.500000  0.500000 )
      -0.0484324491  -0.0484324491    0.1258095002    0.1258095002
      0.2619554320   0.2619554320    0.6015285289    0.6015285289
=== energy_eigen_values ===
ik =    2 ( 0.000000  0.490000  0.490000 )
      -0.0540717117  -0.0427149546    0.1258687813    0.1258687813
      0.2607026827   0.2633829946    0.6006244013    0.6006244013
=== energy_eigen_values ===
ik =    3 ( 0.000000  0.480000  0.480000 )
      -0.0596299923  -0.0369220783    0.1260465996    0.1260465996
      0.2596226501   0.2649874134    0.5980547648    0.5980547648
=== energy_eigen_values ===
ik =    4 ( 0.000000  0.470000  0.470000 )
      -0.0651046420  -0.0310567694    0.1263428799    0.1263428799
      0.2587131916   0.2667706685    0.5941566835    0.5941566835
=== energy_eigen_values ===
ik =    5 ( 0.000000  0.460000  0.460000 )
      -0.0704931128  -0.0251220735    0.1267574962    0.1267574962
      0.2579721226   0.2687346642    0.5892968047    0.5892968047

```

最初の 2 行は、それぞれ、 k 点とバンドの数を表します。3 行目は、この計算でスピン分極は考慮されていないことを、また、4 行目は価電子帯上端におけるエネルギーの値を指しています。

ツール **dos.pl** を使って、電子状態密度の図を作成します。描画するエネルギー範囲の最小値 **E1** と最大値 **E2**

を決めて、

```
% dos.pl dos.data -erange=E1,E2 -color
```

とすると、Postscript 形式の状態密度図 `density_of_states.eps` が得られます。また、`-with_fermi` というオプションをつけて、この処理を実行すると、生成される状態密度図にフェルミ・レベルが点線で描かれます。ただし、ギャップのある系では、価電子帯のエネルギー最大値のところに点線が引かれます。この例題では、以下のように実行します。

```
% dos.pl dos.data -erange=-13,5 -with_fermi -color
```

Si₂の状態密度を、図 4-5 に示します。

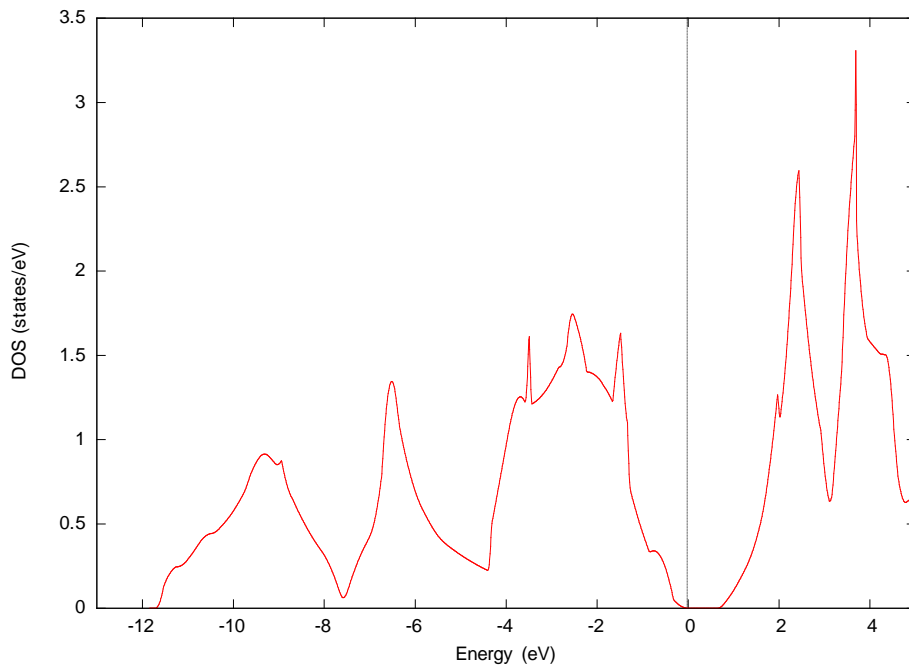


図 4-5 Si₂ の状態密度

(3) バンド構造図

状態密度 (DOS) を計算します。計算例題は、`sample/Si2/band` です。

`file_names.data` では、入出力ファイルを以下のように指定しています。

```
F_INP      = './input_band_Si.data'
F_POT(1)   = '../..pp/Si_ldapw91_nc_01.pp'
F_KPOINT   = '../tools/kpoint.data'
F_CHGT     = '../scf/nfchgt.data'
...        ...
```

入力ファイルは `input_band_Si.data` を、k 点のデータは `kpoint.data` であることを指定しています。

入力ファイル `kpoint.data` は、ツール `band_kpoint.pl` を用いて生成します。生成する k 点をファイル `bandkpt_fcc_xglux.in` で指定します。

```
% band_kpoint.pl bandkpt_fcc_xglux.in
```

これらの入力ファイルを使って、プログラム `ekcal` を実行します。

```
% mpirun ../..bin/ekcal
```

出力ファイル `nfenergy.data` から、ツール `band.pl` を用いて、バンド構造図を作成します。

ツール `band.pl` を以下のように実行すると、Postscript 形式のファイル `band_structure.eps` が作成されます。

```
% band.pl nfenergy.data bandkpt_fcc_xglux.in -erange=E1,E2 -with_fermi -color
```

この例題では、描画するエネルギー範囲の最小値 $E1$ と最大値 $E2$ を、以前同様 $E1=-13$ と $E2=5$ として、以下のように実行します。

```
% band.pl nfenergy.data bandkpt_fcc_xglux.in -erange=-13,5 -with_fermi -color
```

Si2 のバンド構造を、図 4-6 に示します。

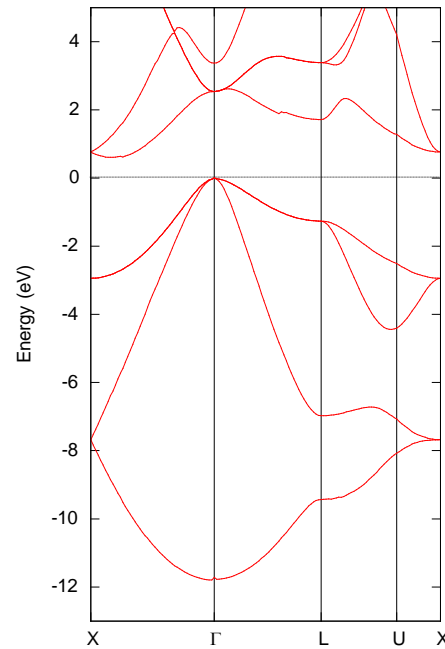


図 4-6 Si2 のバンド構造

4.3 スピン分極を考慮した計算

強磁性体や反強磁性体を扱う場合にはスピン分極を考慮する必要があります。スピン分極の考慮した計算について説明します。

ここでは、強磁性の例として体心立方鉄を、反強磁性の例として体心立方クロムを利用して説明を行います。

4.3.1 強磁性の計算

4.3.1.1 入力パラメータ

強磁性の例として体心立方鉄を例に説明します。計算例題は、`sample/bcc_Fe` です。

```
Control{
  condition = initial
  cpumax = 3 hour
  max_iteration = 250
}

accuracy{
  cutoff_wf = 25 rydberg
  cutoff_cd = 225.00 rydberg
  num_bands = 20
  ksampling{
    method = mesh
    mesh{ nx = 10, ny = 10, nz = 10 }
  }
  smearing{
    method = tetrahedral
  }
  xctype = ggapbe
  scf_convergence{
    delta_total_energy = 1.e-10 hartree
    succession = 3
  }
}

structure{
  unit_cell_type = Bravais
  unit_cell{
    #units angstrom
    a = 2.845, b = 2.845, c = 2.845
    alpha = 90, beta = 90, gamma = 90
  }

  symmetry{
    crystal_structure = bcc
  }

  magnetic_state = ferro

  atom_list{
    atoms{
      !#tag rx ry rz element
      0.000 0.000 0.000 Fe
    }
  }
}
```



```

    }
    element_list{ !#tag element   atomicnumber      zeta   dev
                  Fe              26      0.275  1.5 }
}

Postprocessing{
  dos{
    sw_dos = ON
    method = tetrahedral
    deltaE = 1.e-4 hartree
    nwd_dos_window_width = 10
  }
  charge{
    sw_charge_rspace      = OFF
    filetype               = cube
    title = "This is a title line for FM bcc Fe"
  }
}

printlevel{
  base = 1
}

```

(1) 結晶構造の指定

変数 `crystal_structure` で、体心立方構造の結晶(`bcc` という値)であることを指定しています。よって、ユニットセルはブラベー格子によって指定しているので、原子は 1 つのみ記述しています。体心位置にある原子は指定していない点にご注意ください。 `crystal_structure` に `bcc` という値を指定すると、プログラムが指定の格子を基本格子に変換するので、体心位置の原子の指定は不要となります。

(2) スピン自由度の指定方法

強磁性体を扱う場合には、`magnetic_state` を `ferro` と指定します。

```

structure{
  magnetic_state = ferro  !{para|antiferro|ferro}
}

```

さらに、各原子のスピン分極の初期値を指定する必要があります。入力ファイルにある、

```

    element_list{ #tag element   atomicnumber      zeta   dev
                  Fe              26      0.275  1.5
    }

```

の `zeta=0.275` という変数の値が、アップ・スピンとダウン・スピンの 密度の差を表す、スピン分極 $\zeta = (n_{\uparrow} - n_{\downarrow}) / (n_{\uparrow} + n_{\downarrow})$ の初期値を示しています。

4.3.1.2 計算結果の出力

スピン分極の変化は、ログファイル `output000` に出力されます。以下のようにして確認することができます。

```
% grep charge output000 | grep NEW | more
```

```

!*--- input-file style = NEW
!NEW total charge (UP, DOWN, SUM) =      4.91749982 (+)      3.08250018 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.75677803 (+)      3.24322197 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.64472738 (+)      3.35527262 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.55104317 (+)      3.44895683 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.47221206 (+)      3.52778794 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.46057861 (+)      3.53942139 (=)      8.00000000

```

```

!NEW total charge (UP, DOWN, SUM) =      4.48476557 (+)      3.51523443 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.52141098 (+)      3.47858902 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.56555794 (+)      3.43444206 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.61364243 (+)      3.38635757 (=)      8.00000000
.....
.....
.....

!NEW total charge (UP, DOWN, SUM) =      5.11286684 (+)      2.88713316 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      5.11285665 (+)      2.88714335 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      5.11284790 (+)      2.88715210 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      5.11284030 (+)      2.88715970 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      5.11283035 (+)      2.88716965 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      5.11282059 (+)      2.88717941 (=)      8.00000000

```

ここで、スピン分極の定義 $\zeta = (n_{\uparrow} - n_{\downarrow}) / (n_{\uparrow} + n_{\downarrow})$ を使うと、これが $\zeta = 0.2782$ という値に収束していることが分かります。

以下のように実行すると、更新の前後での電荷分布の変化が確認できます。

```

% grep charge output000 | more

F_CHGT      = ./nfcharge.data                                opened = false
!*** --- charge preconditioning ---
!*** sw_charge_rspace      =      0
!*** charge_filetype      =      1
!*** charge_title          =
!*** deviation( 1) of the Gauss. distrib. func. for the initial charge construction =      1.50000
F_CHGT      = ./nfcharge.data
F_CHGT      = ./nfcharge.data
!! total_charge =      8.000000 (m_CD_initial_CD_by_Gauss_func)
!OLD total charge (UP, DOWN, SUM) =      5.10000000 (+)      2.90000000 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.91749982 (+)      3.08250018 (=)      8.00000000
!OLD total charge (UP, DOWN, SUM) =      4.91749982 (+)      3.08250018 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.75677803 (+)      3.24322197 (=)      8.00000000
!OLD total charge (UP, DOWN, SUM) =      4.75677803 (+)      3.24322197 (=)      8.00000000
!NEW total charge (UP, DOWN, SUM) =      4.64472738 (+)      3.35527262 (=)      8.00000000
.....
.....
.....

```

4.3.2 反強磁性の計算

反強磁性体の場合も、強磁性の計算と基本的には同じです。ただし、反強磁性を実現するためには初期スピン配置を反強磁性的にする必要があります。 そうしないと、高い確率で準安定状態である強磁性の解へ収束します。 強磁性の項においても説明したように、初期スピン分極は元素ごとにしか定義することができません。そこで、**PHASE** では同じ擬ポテンシャルを利用する元素を複数用意し、各々にスピン分極を設定することによって反強磁性的な初期スピン配置を指定することができます。

4.3.2.1 入力パラメータ

反強磁性の例として体心立方クロムを例として説明します。

ここでは、反強磁性秩序をスピン分極が異なる原子を異なる原子種として扱う(磁気秩序 **magnetic_state** は **ferro** と指定する) 方法を紹介します。 **Cr** の元素指定は、以下のように **Cr1** と **Cr2** として指定します。

```

element_list{
  #tag element  atomicnumber  zeta
    Cr1          24          0.3
    Cr2          24         -0.3
}
}

```

Cr1 と Cr2 という 2 種の元素を定義し、初期スピン分極としてそれぞれ 0.3, -0.3 という値を設定しています。原子座標は次のように設定します。これは初期値で、電子状態計算が進むに従いスピン分極の大きさはこの設定値から変化することに注意して下さい。

```
atom_list{
  atoms{
    #tag  rx      ry      rz      element
          0.000    0.000    0.000        Cr1
          0.500    0.500    0.500        Cr2
  }
}
```

原点位置の原子 Cr1 に、体心位置の原子を Cr2 にしています

スピン自由度の指定として、magnetic_state を ferro と指定します。

```
magnetic_state = ferro    !{para|ferro}
```

file_names.data ファイルでは、擬ポテンシャルを次のように指定します。

```
&fnames
F_INP      = './nfinp.data'
F_POT(1)   = '../..Cr_paw1.pp'
F_POT(2)   = '../..Cr_paw2.pp'
/
```

Cr_paw1.pp と Cr_paw2.pp は、内容としては同じ擬ポテンシャルファイルです。これによって、Cr1, Cr2 も同じ擬ポテンシャルが使用されることになります。

この方法を利用することによって、より複雑な磁気構造を持つ系の計算を行うことも可能です。

4.4 構造最適化

原子に働く力を利用して、構造最適化を行うことができます。構造最適化機能の利用方法を説明します。

4.4.1 入力パラメータ

構造最適化を行うには、入力ファイルを次のように記述します。

accuracy ブロックにおいて原子に働く力の最大値の指定を以下のように行います。このパラメータが、構造最適化の収束判定となります。

```
...
accuracy{
    ...
    max_force = 1.0e-3 hartree/bohr
    ...
}
...
```

max_force のデフォルト値は、 10^{-3} hartree/bohr です。

Structure ブロックの原子の指定 **atom_list** に **mobile** 属性を定義し、最適化の対象となる原子に 1 という値を指定します。最適化の対象としない原子は 0 あるいは*とします。

```
...
structure{
    ...
    atom_list{
        !#tag element rx ry rz mobile
        Ba 0.0000 0.5000 0.05 0
        O 0.5000 0.0000 0.05 1
        Ba 0.5000 0.0000 0.15 1
        O 0.0000 0.5000 0.15 1
        ...
    }
}
...
```

この例では、1 番目の Ba 原子は最適化の対象とせず、2 番目と 4 番目の O 原子と 3 番目の Ba 原子が最適化の対象としています。

x, *y*, *z* 座標を個別に最適化の対象とするかどうかを設定することも可能です。この設定は、**mobilex**, **mobile**, **mobilez** 属性値によって行います。**mobilex**, **mobile**, **mobilez** 属性値は、**mobile** 属性値と同じ値がデフォルト値です。

```
...
structure{
    ...
    atom_list{
        !#tag element rx ry rz mobile mobilex
        Ba 0.0000 0.5000 0.05 0 1
        O 0.5000 0.0000 0.05 1 0
        Ba 0.5000 0.0000 0.15 1 1
        O 0.0000 0.5000 0.15 1 0
        ...
    }
}
...
```

この例では、1 番目の Ba 原子は*x* 座標のみが、2 番目および 4 番目の O 原子は*y* 座標と *z* 座標のみが、3 番目の Ba 原子は*x*, *y*, *z* 座標が最適化の対象となります。

structure_evolution ブロックに、構造最適化の設定をします。

```
...
structure_evolution{
    method = quench
    dt = 50
    ...
}
...
```

method 構造緩和の方法を指定します。構造緩和のオプションとして、quench (quenched MD 法), cg (CG 法), cg2 法 (改良 CG 法) gdiis (GDIIS 法), bfgs (BFGS 法) のいずれかが選べます。デフォルト値は bfgs です。

dt 構造緩和を行う際の時間刻みです。大きい方が早く収束へいたりしますが、大きすぎると計算を正しく進行させることができなくなる場合があります。デフォルト値は原子単位で 100 です。

GDIIS 法あるいは BFGS 法は原子に働く力が大きい場合安定に計算できない場合があるので、力が大きい内は quenched MD 法か CG 法を利用し、ある程度力が小さくなってから GDIIS(BFGS)法に切り替える、という動作をします。GDIIS(BFGS)に切り替える前の最適化手法と切り替えの判定条件は、それぞれ変数 initial_method と c_forc2gdiis を利用して 次のように設定します。

```
...
structure_evolution{
    method = gdiis
    dt = 50
    gdiis{
        initial_method = cg
        c_forc2gdiis = 0.0025 hartree/bohr
    }
}
...
```

ブロック名は、GDIIS, BFGS 共通で gdiis です。デフォルト値は initial_method が cg, c_forc2gdiis が 0.05 hartree/bohr です。

4.4.2 計算結果の出力

構造最適化を施すと、F_ENF ファイル(既定のファイル名: nfefn.data)にエネルギーや原子に働く力の最大値の履歴が、F_DYNM ファイル(既定のファイル名: nfdynm.data)に原子配置の履歴が出力されます。

4.4.3 計算例: シリコン結晶の構造最適化

シリコン結晶の構造最適化の計算例です。安定な原子配置から原子位置をずらして、そこからの緩和過程を計算する例題です。計算例題は、sample/Si2/relax です。

(1) 入力ファイル

ファイル file_names.data の中では、入力ファイル input_relax_Si.data と、原子の位置座標と各原子に働く力の計算結果の出力ファイル nfdynm.data が指定されています。

```
F_INP      = './input_relax_Si.data'
...
F_DYNM     = './nfdynm.data'
```

...

入力ファイル `input_relax_Si.data` は、格子間隔を 0.125 ではなく 0.130 とし、安定な原子配置から原子位置をずらしています。また、`mobile` 変数の値を `yes` にして、原子位置を可変にしています。

```
structure{
  ...
  atom_list{
    atoms{
      #tag      rx      ry      rz      element mobile
              0.130   0.130   0.130   Si      yes
              -0.130  -0.130  -0.130  Si      yes
    }
  }
}
```

`accuracy` ブロックで原子に働く力の収束条件を指定します。

```
accuracy{
  force_convergence{
    max_force = 1.0e-3
  }
}
```

(2) 計算結果

計算結果の出力ファイル `nfdynm.data` は以下の通りです。

```
#
# a_vector =      0.0000000000      5.1300000000      5.1300000000
# b_vector =      5.1300000000      0.0000000000      5.1300000000
# c_vector =      5.1300000000      5.1300000000      0.0000000000
# ntyp =      1 natm =      2
# (natm->type)      1      1
# (speciesname)      1 :      Si
#
cps and forc at (iter_ion, iter_total =      1      34 )
  1  1.333800000  1.333800000  1.333800000  -0.010794  -0.010794  -0.010794
  2 -1.333800000 -1.333800000 -1.333800000   0.010794   0.010794   0.010794
cps and forc at (iter_ion, iter_total =      2      53 )
  1  1.331707297  1.331707297  1.331707297  -0.010402  -0.010402  -0.010402
  2 -1.331707297 -1.331707297 -1.331707297   0.010402   0.010402   0.010402
cps and forc at (iter_ion, iter_total =      3      75 )
  1  1.327597870  1.327597870  1.327597870  -0.009614  -0.009614  -0.009614
  2 -1.327597870 -1.327597870 -1.327597870   0.009614   0.009614   0.009614
cps and forc at (iter_ion, iter_total =      4     100 )
  1  1.321624355  1.321624355  1.321624355  -0.008433  -0.008433  -0.008433
  2 -1.321624355 -1.321624355 -1.321624355   0.008433   0.008433   0.008433
cps and forc at (iter_ion, iter_total =      5     127 )
  1  1.314015753  1.314015753  1.314015753  -0.006865  -0.006865  -0.006865
  2 -1.314015753 -1.314015753 -1.314015753   0.006865   0.006865   0.006865
cps and forc at (iter_ion, iter_total =      6     155 )
  1  1.305076108  1.305076108  1.305076108  -0.004930  -0.004930  -0.004930
  2 -1.305076108 -1.305076108 -1.305076108   0.004930   0.004930   0.004930
cps and forc at (iter_ion, iter_total =      7     184 )
  1  1.295180554  1.295180554  1.295180554  -0.002671  -0.002671  -0.002671
  2 -1.295180554 -1.295180554 -1.295180554   0.002671   0.002671   0.002671
cps and forc at (iter_ion, iter_total =      8     213 )
  1  1.284767108  1.284767108  1.284767108  -0.000159  -0.000159  -0.000159
  2 -1.284767108 -1.284767108 -1.284767108   0.000159   0.000159   0.000159
```

このうち、`#` 記号で始まる部分は入力データの一部を表していますが、その次の行は、イオンすなわちコア原子の位置座標を一回更新する間に、全更新回数が 34 回であったこと、すなわち、この間に波動関数が 33 回更新されたことを示しています。波動関数の更新に対する収束条件は、第 3 節の例題と同様に、全エネルギーに

対して課されています。

また、その次の 2 行は、原子の番号、原子位置(x,y,z, bohr 単位)、 および力の成分 (x,y,z, hartree/bohr 単位)の計算結果を表しています。これにより、結果を下まで辿ってゆくと、計算が進むにつれて、原子に働く力が急激に減少してゆくことが分かります。最後の更新で、力の各成分の計算結果が、最初に指定された収束条件以下になったために、緩和過程の計算が終了しています。

4.5 表面の計算

4.5.1 表面の計算を実行するには

PHASE は系に周期境界条件を課す必要があるのですが、厳密な意味では表面などの有限系を扱うことはできません。しかし、十分な“真空層”を設けることにより、事実上表面と変わらない系を扱うことは可能です。真空層は、底面と表面が相互作用しない程度の大きさを取ります。通常、 10\AA 以上の真空層を採用します。水素終端されたシリコン表面の計算を例とします。この構造の計算には、図 4-7 に示されるようなスラブ模型を使います。スラブの下側の Si 原子のボンドは、仮想的な水素原子で終端しています。

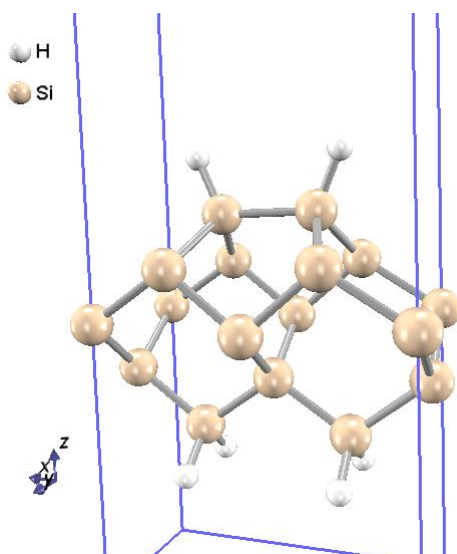


図 4-7 水素終端した Si(001)-p(2×1) 表面の構造図

この例で使用する file names.data です。

```
&fnames
  F_INP      = './input_SiH2x1.data'
  F_POT(1)   = '../pp/Si_ldapw91_nc_01.pp'
  F_POT(2)   = '../pp/H_ldapw91_nc_01.pp'
  .....
&end
```

F_POT(1) と F_POT(2) に、Si 原子と H 原子の擬ポテンシャルを指定しています。

入力パラメータ例です。

k 点サンプリングの指定です。

```
accuracy{
  cutoff_wf = 15.00 rydberg
  cutoff_cd = 60.00 rydberg
  num_bands = 25
  ksampling{
    method = monk ! {mesh|file|directin|gamma}
    mesh{ nx = 2, ny = 4, nz = 1 }
    kshift{ k1 = 0.5, k2 = 0.5, k3 = 0.0 }
  }
  .....
}
```


この例では、スラブ模型を用いているため、 k 点のサンプリングは、 k_z 方向には1点だけを取っています。

```

structure{
  unit_cell_type = primitive
  unit_cell{
    a_vector = 14.512      0.000      0.000
    b_vector =  0.000      7.256      0.000
    c_vector =  0.000      0.000     30.784
  }
  symmetry{}

  magnetic_state = para    !{para|af|ferro}

  atom_list{
    coordinate_system = internal
    atoms{
      #default weight = 1, element = Si, mobile = 0
      #tag    rx      ry      rz      element
      0.26177 0.50000 0.65651      H
      0.73823 0.50000 0.65643      H
      0.34138 0.50000 0.56971
      0.65858 0.50000 0.56966
      0.26229 0.00000 0.49388
      0.73763 0.00000 0.49385
      0.00000 0.00000 0.41498
      0.50000 0.00000 0.40298
      0.00000 0.50000 0.32769
      0.50000 0.50000 0.32150
      0.25000 0.50000 0.24167
      0.75000 0.50000 0.24167
      0.25000 0.20000 0.18269      H
      0.25000 0.80000 0.18269      H
      0.75000 0.20000 0.18269      H
      0.75000 0.80000 0.18269      H
    }
  }
}

postprocessing{
  charge{
    sw_charge_rspace = ON
    filetype = cube !{cube|density_only}
    title = "Si(001) p(2x1) surface terminated by H atoms"
  }
}

```

atoms の中で、デフォルト値として元素名を Si に設定しているので、変数 element に H と入力している以外の原子の元素名は Si になります。また、やはりデフォルト値として mobile=0 としているため、全ての原子の座標位置は固定されています。

```
% grep TOTAL output000
```

として、全エネルギーの収束状況を確認すると、以下のような結果が得られます。

TOTAL ENERGY FOR	1 -TH ITER=	-41.206501960258	edel =	-0.412065D+02	: SOLVER =	MATDIAGON
TOTAL ENERGY FOR	2 -TH ITER=	-42.928541839902	edel =	-0.172204D+01	: SOLVER =	DAVIDSON
TOTAL ENERGY FOR	3 -TH ITER=	-42.956734520103	edel =	-0.281927D-01	: SOLVER =	DAVIDSON
TOTAL ENERGY FOR	4 -TH ITER=	-42.960659333525	edel =	-0.392481D-02	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	5 -TH ITER=	-42.961623666220	edel =	-0.964333D-03	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	6 -TH ITER=	-42.962559338199	edel =	-0.935672D-03	: SOLVER =	SUBMAT + RMM3
TOTAL ENERGY FOR	7 -TH ITER=	-42.964136746929	edel =	-0.157741D-02	: SOLVER =	SUBMAT + RMM3

TOTAL ENERGY FOR	8 -TH ITER=	-42.964791285123	edel =	-0.654538D-03 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	9 -TH ITER=	-42.964953052183	edel =	-0.161767D-03 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	10 -TH ITER=	-42.965045860995	edel =	-0.928088D-04 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	11 -TH ITER=	-42.965076083146	edel =	-0.302222D-04 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	12 -TH ITER=	-42.965088896548	edel =	-0.128134D-04 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	13 -TH ITER=	-42.965091550789	edel =	-0.265424D-05 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	14 -TH ITER=	-42.965092402734	edel =	-0.851945D-06 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	15 -TH ITER=	-42.965092972980	edel =	-0.570245D-06 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	16 -TH ITER=	-42.965093291397	edel =	-0.318417D-06 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	17 -TH ITER=	-42.965093454357	edel =	-0.162961D-06 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	18 -TH ITER=	-42.965093580068	edel =	-0.125710D-06 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	19 -TH ITER=	-42.965093601039	edel =	-0.209711D-07 : SOLVER = SUBMAT + RMM3
TOTAL ENERGY FOR	20 -TH ITER=	-42.965093604435	edel =	-0.339656D-08 : SOLVER = SUBMAT + RMM3

この例題は、固体表面構造に対するエネルギー計算だけを目的にしていますが、もし原子位置の緩和過程の計算を行う場合は、以下のように、下端の仮想水素とそれらと結合した Si 原子を固定し、それら以外の原子を可動 (mobile = 1) に変えてやる必要があります。

```
atoms{
  #default weight = 1, element = Si, mobile = 1
  #tag    rx      ry      rz      element  mobile
        0.26177  0.50000  0.65651      H
        0.73823  0.50000  0.65643      H
        0.34138  0.50000  0.56971
        0.65858  0.50000  0.56966
        0.26229  0.00000  0.49388
        0.73763  0.00000  0.49385
        0.00000  0.00000  0.41498
        0.50000  0.00000  0.40298
        0.00000  0.50000  0.32769
        0.50000  0.50000  0.32150
        0.25000  0.50000  0.24167      *      0
        0.75000  0.50000  0.24167      *      0
        0.25000  0.20000  0.18269      H      0
        0.25000  0.80000  0.18269      H      0
        0.75000  0.20000  0.18269      H      0
        0.75000  0.80000  0.18269      H      0
}
```

Si(001) 表面のバックルしたダイマーの安定構造は $p(2 \times 1)$ ではなく $c(4 \times 2)$ ですが、この構造を再現するには、Si ダイマーをもう一つ増やすなどして、最上層に位置する Si ダイマーの総数を偶数個にしなければなりません。

4.5.2 反転対称性を考慮した表面の計算

表面には、反転対称性がある場合があります。反転対称性を利用することによって、ほぼ同等の計算負荷で 2 倍の厚さの表面モデルを取り扱うことが可能です。Pt 表面の(111) 面を例とします。この例題の入力ファイルの structure ブロックは以下になっています。

```
structure{
  element_list{
    #tag element atomicnumber mass zeta deviation
    Pt 78 355606.909 0.0 1.83
  }
  atom_list{
    coordinate_system = cartesian
    atoms{
      #units angstrom
      #tag element rx ry rz mobile weight
Pt 0.00 0.00 0.00
```

```

Pt 1.4142135624 2.4494897428 0.00
Pt 2.8284271248 0.00 0.00
Pt 4.2426406871 2.4494897428 0.00
Pt 5.6568542497 3.2659863239 2.30940111
Pt 4.2426406874 0.8164965811 2.30940111
Pt 2.828427125 3.2659863239 2.30940111
Pt 1.4142135626 0.8164965811 2.30940111
Pt 2.8284271245 1.6329931617 4.618802187
Pt 4.2426406868 4.0824829045 4.618802187
Pt 5.6568542492 1.6329931617 4.618802187
Pt 7.0710678116 4.0824829045 4.618802187
Pt 5.6568543525 0.0000002214 6.928203264
Pt 1.4142137683 2.4494897428 6.928203264
Pt 2.8284271248 0.00 6.928203264
Pt 4.2426406871 2.4494897428 6.928203264
Pt 5.6568542497 3.2659863239 9.237604341
Pt 4.2426406874 0.8164965811 9.237604341
Pt 2.828427125 3.2659863239 9.237604341
Pt 1.4142135626 0.8164965811 9.237604341
Pt 2.8284271245 1.6329931617 -2.30940111
Pt 4.2426406868 4.0824829045 -2.30940111
Pt 5.6568542492 1.6329931617 -2.30940111
Pt 7.0710678116 4.0824829045 -2.30940111
Pt 5.6568542497 3.2659863239 -4.618802187
Pt 4.2426406874 0.8164965811 -4.618802187
Pt 2.828427125 3.2659863239 -4.618802187
Pt 1.4142135626 0.8164965811 -4.618802187
Pt 2.8284270217 4.8989792642 -6.928203264
Pt 7.0710676059 2.4494897428 -6.928203264
Pt 2.8284271248 0.00 -6.928203264
Pt 4.2426406871 2.4494897428 -6.928203264
Pt 2.8284271245 1.6329931617 -9.237604341
Pt 4.2426406868 4.0824829045 -9.237604341
Pt 5.6568542492 1.6329931617 -9.237604341
Pt 7.0710678116 4.0824829045 -9.237604341
    }
  }
  unit_cell{
    #units angstrom
    a_vector = 5.6568542495 0.00 0.00
    b_vector = 2.8284271247 4.8989794856 0.00
    c_vector = 0.00 0.00 30.00
  }
  symmetry{
    method = automatic
    tspace{
      lattice_system = primitive
    }
    sw_inversion = on
  }
}

```

この例題では、原点を中心に反転対称性があります。この点を活用するために、**symmetry** ブロックの下の **sw_inversion** 変数を on としています。この座標データを可視化すると、図 4-8 となります。

この例のように、表面は厚さ方向の中央を原点とすることによって反転対称性があるようになる場合があります。このような場合は、**sw_inversion** パラメータを on とすることによって計算量を削減することができます。表面にさらに分子や原子などを吸着させた計算を行う場合は、両側の反転対称位置に配置することによってやはり反転対称性を持たせることが可能です。

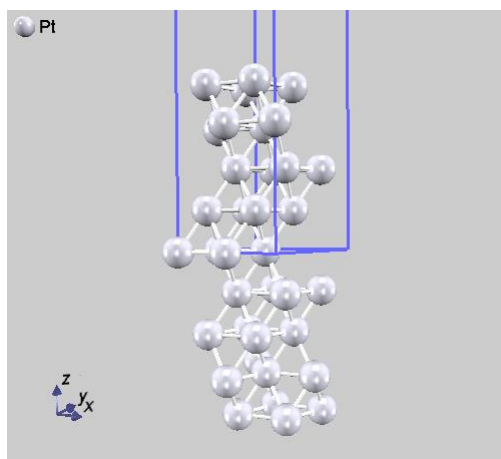


図 4-8 Pt(111) 面の原子配置。表面モデルの中央を原点にすることによって反転対称性がある。

4.5.3 計算例：金属表面の生成エネルギー

0K における表面の生成エネルギーは、以下のように評価することが可能です。

$$\gamma = (E_s - E_b)/2A$$

ここで γ が表面生成エネルギー、 E_s が表面の全エネルギー、 E_b が対応する結晶の全エネルギー、 A が表面積です。 $2A$ で割っているのは、計算では表面が2つ現れるからです。また、 E_b は表面モデルと原子数が合うようにスケールしたあとで差を評価します。

反転対称性を考慮した表面の計算例は、白金表面の生成エネルギーの計算です。

Pt(111) 面	9層の(111)面, 計 36 原子。 格子定数は, $a = b = 5.657\text{\AA}$, $c = 30\text{\AA}$, $\alpha = \beta = 90^\circ$, $\gamma = 120^\circ$ 図 4-8 のモデル
Pt(110)面 MR	15層の missing-row (MR) (110)面, 計 28 原子 MR 面とは, 表面の“列”をなしている原子が 1 列おきに欠けている表面のモデル。 格子定数は $a = 4\text{\AA}$, $b = 2.828427125\text{\AA}$, $c = 30\text{\AA}$, $\alpha = \beta = \gamma = 90^\circ$ 図 4-10 のモデル (この図では, スーパーセルで表示している)。
Pt(110)面	15層の(110)面, 計 15 原子。 格子定数は $a = 8\text{\AA}$, $b = 2.8284271248\text{\AA}$, $c = 30\text{\AA}$, $\alpha = \beta = \gamma = 90^\circ$ 図 4-9 のモデル (この図では, スーパーセルで表示している)。

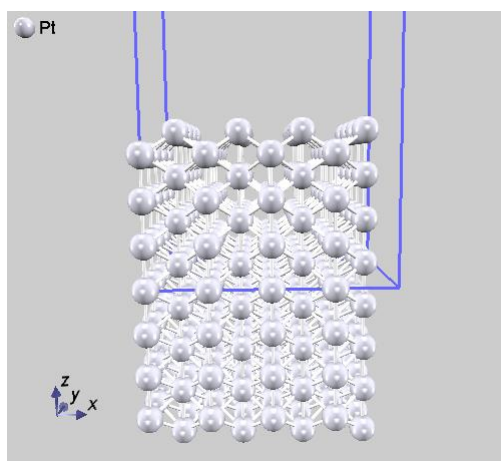


図 4-9 Pt(110) 理想表面(スーパーセル表示)

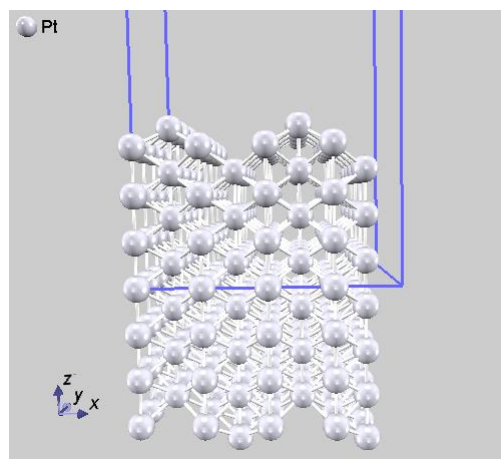


図 4-10 Pt(110) 面 missing-row 構造(スーパーセル表示)

白金表面は, (111)面が最も安定で, (110)面については missing-row 再配列が成されるとされています。このようなことが, 表面生成エネルギーの計算から再現できることを確認します。

主な計算条件です。

- いずれのモデルも反転対称性を考慮
- カットオフエネルギーは 25 Rydberg
- k 点サンプリングは, (111) に対しては $6 \times 6 \times 1$, (110) に対しては $6 \times 8 \times 1$, (110) MR に対しては $3 \times 8 \times 1$
- 構造最適化は BFGS 法によって実施 ; 力の収束判定は 2×10^{-4} hartree/bohr
- 構造最適化の対象となる原子は, 最表面から 4 層ずつ

このようにして得られた表面生成エネルギーの計算結果を, 表 4.3 にまとめました。(111)面の生成エネルギーが小さく, 次に(110) MR, 最も生成エネルギーが大きいのが(110)面という結果が得られました。

表 4.3 白金表面の生成エネルギー。(111), (110)MR, (110) の順で生成エネルギーが小さい。

	(111)	(110) MR	(110)
生成エネルギー (eV/Å ²)	0.089	0.099	0.108

4.6 原子・分子の計算

原子・分子の計算は、真空層を設けることによって行います。原子や分子の場合は、周期的境界条件の影響がないように、すべてのセルベクトルの方向で真空層を設ける必要があります。通常、 k サンプリングは Γ 点のみを利用します。

4.6.1 入力パラメータ

原子・分子の計算は、真空層を設けるように `unit_cell` を指定します。

```
unit_cell{
    a_vector = 15.0      0.0      0.0
    b_vector =  0.0      15.0      0.0
    c_vector =  0.0      0.0      15.0
}
```

水分子の計算の入力パラメータです。原子座標に対し、十分に大きなユニットセルとしています。

```
Control{
    condition = initial
    cpumax = 1 day ! maximum cpu time
    max_iteration = 6000
}

accuracy{
    cutoff_wf = 25.00 rydberg
    cutoff_cd = 225.00 rydberg
    num_bands = 8
    xctype = ggapbe
    initial_wavefunctions = matrix_diagon
    matrix_diagon {
        cutoff_wf = 5.0 rydberg
    }
    ksampling{
        method = gamma
    }
    scf_convergence{
        delta_total_energy = 1.e-10
        succession = 3
        num_max_iteration = 300
    }
    force_convergence{
        max_force = 1.e-4
    }
    initial_charge_density = Gauss
}

structure{
    unit_cell_type = primitive
    unit_cell{
        a_vector = 15.0      0.0      0.0
        b_vector =  0.0      15.0      0.0
        c_vector =  0.0      0.0      15.0
    }
    symmetry{
        tspace{
            lattice_system = primitive
            generators{
```

```

        #tag rotation tx ty tz
        C2z      0 0 0
        IC2x     0 0 0
    }
}

atom_list{
    coordinate_system = cartesian
    atoms{
        !#default mobile=on
        !#tag rx ry rz element
        -1.45 0.000 1.123 H
        1.45 0.000 1.123 H
        0.0 0.0 0.0 O
    }
}

element_list{ #units atomic_mass
    #tag element atomicnumber zeta dev
    H 1 1.00 0.5
    O 8 0.17 1.0 }
}

wf_solver{
    solvers {
        !#tag sol till_n dts dte itr var prec cmix submat
        msd 5 0.1 0.1 1 tanh on 1 on
        lm+msd 10 0.1 0.4 50 tanh on 1 on
        rmm2p -1 0.4 0.4 1 tanh on 2 on
    }
    rmm {
        edelta_change_to_rmm = 1.d-6
    }
    lineminimization {
        dt_lower_critical = 0.1
        dt_upper_critical = 3.0
    }
}

charge_mixing{
    mixing_methods {
        !#tag id method rmxs rmxe itr var prec istr nbxmix update
        1 broyden2 0.3 0.3 1 linear on 5 10 RENEW
        2 simple 0.2 0.5 100 linear on * * *
    }
}

```

4.7 電荷密度の出力

PHASE は SCF 計算中は逆空間で電荷密度を扱いますが、収束した電荷密度を実空間に逆フーリエ変換し、出力させることも可能です。こうすることによって PHASE-Viewer などを利用して電荷密度の可視化を行うことが可能です。電荷密度を実空間に出力させるためには、入力ファイルの最上位に `postprocessing` ブロックを作成し、さらにその下に `charge` ブロックを作成しその下で設定を行います。

```
postprocessing{
  charge{
    sw_charge_rspace = on
    filetype = cube
  }
}
```

`charge` ブロックの下では以下の変数の設定を行います。

<code>sw_charge_rspace</code>	電荷密度を実空間で出力するかどうかを指定する真偽値です。on にすると実空間の電荷密度が出力されます。
<code>filetype</code>	電荷密度データのデータフォーマットを指定します。 <code>density_only</code> と <code>cube</code> が選べます。 <code>density_only</code> の場合電荷密度のみが出力されます。デフォルト値は <code>density_only</code> です。 <code>cube</code> の場合、Gaussian Cube 形式で電荷密度が出力されます。このパラメーターは、 <code>cube</code> に設定することを推奨します。
<code>title</code>	Gaussian Cube ファイルの“見出し”を指定します。空白文字を含める場合、全体を半角の 2 重引用符で囲みます。

また、`filetype` として `cube` を選択した場合、`file_names.data` ファイルにおいて電荷密度ファイルのファイル名を変更しておくことを推奨します。

```
&fnames
...
F_CHR = './nfchr.cube'
/
```

変更しない場合のデフォルト値は `nfchr.data` です。

スピン分極を考慮している場合は、`file_names.data` で指定したファイル名が `nfchr.cube` であったとすると、`nfchr.up.cube` と `nfchr.down.cube` という 2 つのファイルにそれぞれスピンアップ・ダウン に対応する電荷密度データが出力されます。参考のため、図 4-11 に鉄の多数派スピンと少数派スピンの電荷密度を PHASE-Viewer で可視化した様子を示します。

さらに、特定のエネルギー範囲の電荷密度を抜き出して出力させる機能も PHASE には備わっています。この機能については、応用機能において解説します。

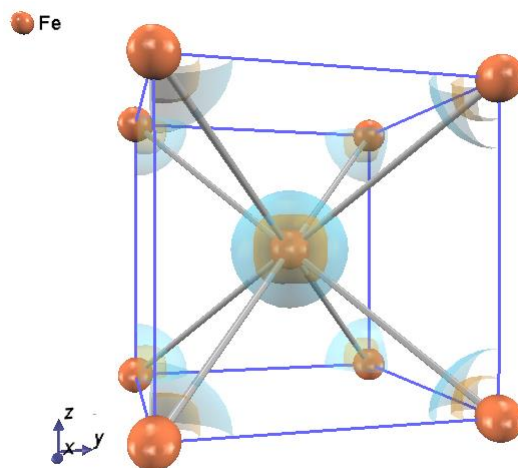


図 4-11 Fe の電荷密度分布図. 青色とオレンジ色の面は, 自発磁化により生じた, 多数派スピンの多数派スピンと少数派スピンの電荷密度分布の等値面を表す.

4.8 状態密度の計算

SCF 計算が収束したのち、状態密度の計算を行わせることができます。電荷密度の計算を行うためには、入力ファイルの最上位に **postprocessing** ブロックを作成し、さらにその下に **dos** ブロックを作成しその下で設定を行います。

```
postprocessing{
  dos{
    sw_dos = on
    method = gaussian
    deltaE_dos = 1e-4 hartree
  }
}
```

dos ブロックでは以下の設定を行うことができます。

sw_dos	状態密度計算を行うかどうかを指定する真偽値です。状態密度の計算を行う場合 on とします。
method	状態密度の計算方法を指定します。 gaussian と tetrahedral のいずれかを選択することができます。 gaussian を選択した場合、エネルギー準位をガウス関数によって幅を持たせた上で計算した状態密度が得られます。 tetrahedral の場合四面体法による高精度な状態密度計算を行うことができます。ただし tetrahedral を利用する場合後述の四面体法が利用できる条件もご参照ください。
deltaE_dos	状態密度計算に利用されるエネルギーの幅をハートリー単位で指定します。デフォルト値は 1e-4 hartree です。

状態密度の計算方法として **tetrahedral** を利用する場合、以下の条件が満たされている必要があります。

- **k** 点サンプリング手法として **mesh** 法を採用している

```
accuracy{
  ksampling{
    method = mesh
  }
}
```

- **smearing** の方法として **tetrahedral** 法を採用している

```
accuracy{
  smearing{
    method = tetrahedral
  }
}
```

以上が満たされていないと **gaussian** 法による状態密度計算が行われてしまうので、ご注意ください。

参考のため、**gaussian** 法と **tetrahedral** 法で計算した体心立方鉄の状態密度をそれぞれ図 2.10 と 2.11 に示します。**k** 点メッシュはそれぞれ $10 \times 10 \times 10$ を採用しました。図 4-12 および図 4-13 より、**Tetrahedral** 法で計算状態密度の方がシャープで精度のよいものが得られていることが分かります。

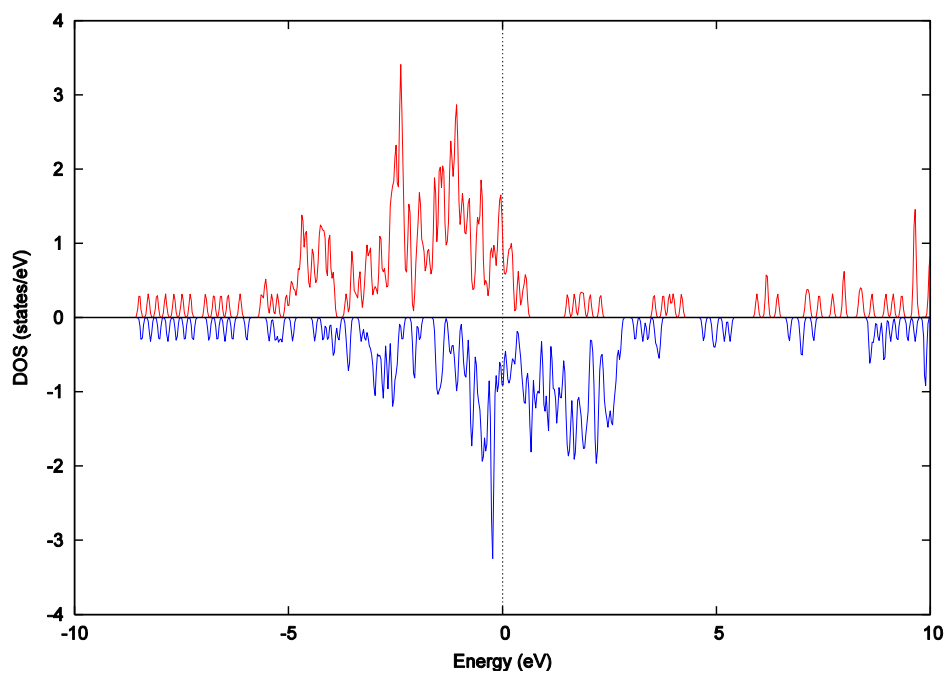


図 4-12 Gaussian 法で計算した体心立方鉄の状態密度

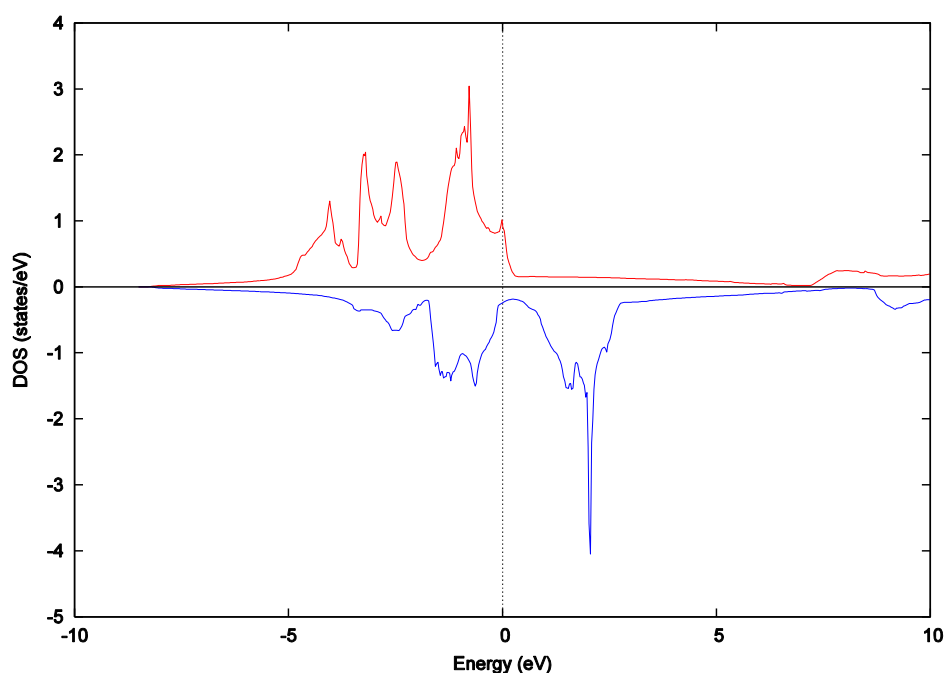


図 4-13 Tetrahedral 法で計算した体心立方鉄の状態密度

PHASE には原子や層によって分割した“局所状態密度”を計算する機能も備わっています。この機能については、応用機能において説明します。

4.9 バンド構造の計算

4.9.1 k 点のデータの作成

バンド構造の計算には、バンド分散を計算する k 点のデータが必要です。

k 点のデータは、ツール `band_kpoint.pl` を利用して作成します。まず `band_kpoint.pl` 用の入力ファイルを作成します。その形式は、以下のようなものです。

```
dkv
b1x b2x b3x
b1y b2y b3y
b1z b2z b3z
n1 n2 n3 nd # Symbol
...
```

`dkv` が k 点の間隔, `b1x,b1y,b1z` は逆格子ベクトル b_1 の x,y,z 成分です。逆格子ベクトル b_2, b_3 についても同様です。五行目以降に特殊 k 点とそのシンボルの指定をします。シンボルの指定は必須ではありませんが、指定がある場合、バンド構造図作成の際に利用されます。整数 n_1, n_2, n_3, n_d を用いて k ベクトルを

$$k = \frac{n_1}{n_d} b_1 + \frac{n_2}{n_d} b_2 + \frac{n_3}{n_d} b_3$$

のように指定します。シンボルは `#` の後に書いてください。面心立方格子の場合の例を示します。

```
0.02          <---- k 点の間隔
-1.0  1.0  1.0
1.0 -1.0  1.0  <---- 逆格子ベクトル
1.0  1.0 -1.0
0 1 1 2 # X      <---- n1 n2 n3 nd # Symbol
0 0 0 1 # {/Symbol G}
1 1 1 2 # L
5 2 5 8 # U
1 0 1 2 # X
```

このファイルを作成したら、以下のように `band_kpoint.pl` を実行すればファイル `kpoint.data` が作成されます。

```
% band_kpoint.pl bandkpt.in
```

`kpoint.data` は以下のような記述になっています。

```
141 141      (a)
0 50 50 100 1 (b)
0 49 49 100 1
0 48 48 100 1
0 47 47 100 1
0 46 46 100 1
0 45 45 100 1
0 44 44 100 1
0 43 43 100 1
.....
.....
.....
```

各項目は次のような意味です。

- (a) k 点の個数を指定します。この例では、141 個です。
- (b) 4つの整数は、それぞれ k 点を次式のように定義した場合の n_1, n_2, n_3, n_d, w になります (ここで $\vec{b}_1, \vec{b}_2, \vec{b}_3$ は逆格子ベクトルです)。

$$\vec{k} = w \times \left(\frac{n_1}{n_d} \vec{b}_1 + \frac{n_2}{n_d} \vec{b}_2 + \frac{n_3}{n_d} \vec{b}_3 \right)$$

4.9.2 固定電荷の計算を行う

固定電荷の計算は、プログラム **ekcal** もしくは **phase** を使用します。

SCF の計算を行ったディレクトリに固定電荷計算を実行しても問題はありませんが、波動関数などのデータが上書きされないようにするため新たに固定電荷用の実行ディレクトリを作成することをお勧めします。

4.9.2.1 入力パラメータ

(1) file_names.data

file_names.data は基本的には SCF 計算の場合と同様ですが、F_CHGT 識別子で SCF 計算によって得られた電荷密度ファイルを指す必要がある点が異なります。このファイルは SCF 計算で利用した file_names.data 中の F_CHGT 識別子で指定されるファイルであり、既定の名前は nfchgt.data です。たとえば、SCF 計算を行ったディレクトリ直下において固定電荷用の入力データを作成している場合、file_names.data に以下を記述します。バンド分散を求めたい k 点のデータ kpoint.data は file_names.data で指定できます。

```
&fnames
...
F_CHGT = '../nfchgt.data'
F_KPOINT = 'kpoint.data'
...
/
```

もし PAW 法による計算を行っているのならば、F_CHGT のほかに F_CNTN_BIN_PAW という識別子によって指定されるファイルも SCF 計算のファイルを指す必要があります。また、DFT+U 法による計算を行っている場合、占有行列ファイルを SCF 計算のファイルを F_OCCMAT 識別子によって指定する必要があります。具体的には、以下のようになります。

```
&fnames
...
F_CHGT = '../nfchgt.data'
F_OCCMAT = '../occmat.data' <--- DFT+U の場合は必要
F_CNTN_BIN_PAW = '../continue_bin_paw.data' <--- PAW 法の場合は必要
...
/
```

(2) 入力パラメータファイル

固定電荷計算用の入力ファイルを作成します。基本的には SCF 計算で利用した入力ファイルを元に作成するとよいでしょう。ただし次の点にご注意いただく必要があります

● 原子の座標の作成

構造緩和を行った場合、固定電荷の入力ではその緩和された構造を利用する必要があります。従って、構造緩和を行った場合は F_DYNM ファイルに書かれている最後の構造を参考に原子の座標を設定してください。

● 計算条件の変更

固定電荷で計算する、という指定を下記の要領で行います。

```
Control{
...
condition = fixed_charge
...
}
```

...

固定電荷の計算も継続計算に対応しています。継続計算を行う場合, `condition` を `fixed_charge_continuation` としてください。

- **k** 点サンプリングの設定

作成した `kpoint.data` を読み込むように, **k** 点サンプリング法を以下のように編集します。

```
accuracy{
  ...
  ksampling{
    method = file
  }
  ...
}
```

- `ek_convergence` ブロックの設定

固有値計算の計算条件などを設定する, `accuracy.ek_convergence` ブロックの設定を行う必要があります。以下, `ek_convergence` ブロックの記述の例を示します。

```
accuracy{
  ...
  ek_convergence{
    num_max_iteration = 500
    delta_eigenvalue = 1.e-5
    succession = 2
  }
  ...
}
```

`ek_convergence` ブロックの各変数の意味は下記の通りです

`num_max_iteration` 繰り返し計算の上限を指定します

`delta_eigenvalue` 収束判定を設定します。この値のデフォルト値は $1e^{-5}$ です。多くの場合このデフォルト値で問題はないと思われますが、目安としては、絶縁体、半導体の場合は $1.e^{-4}$ rydberg 程度、金属の場合は $1.e^{-6}$ rydberg 程度がよいでしょう。

`succession` 全エネルギーの前ステップとの差が `delta_eigenvalue` 以下 `succession` 回連続で収まった時点で収束したと見做します。

(3) **k** 点処理の方法の選択

固定電荷計算に `phase` を用いる場合, **k** 点処理の方法を選択することができます。すなわち, **k** 点を一点ずつ処理するか, すべて同時に処理するかを選ぶことができます。メモリ要求は前者の方が少ないですが, **k** 点並列の効率性は後者の方が高い場合が多いです。 `phase` を使って固定電荷計算を行う場合のデフォルト値はすべて同時に処理する方法です。一点ずつ処理するように動作させるためには, 以下のような設定を施します。

```
control{
  fixed_charge_option{
    kparallel = one_by_one
  }
}
```

4.9.3 バンド構造図の作成

計算を実行した結果、全 k 点の各バンドの固有エネルギーがファイル `nfenergy.data` に出力されます。

```

num_kpoints = 117 (a)
num_bands = 8 (b)
nspin = 1 (c)
Valence band max = 0.233846 (d)

nk_converged = 117 (e)
ik = 1 ( 0.500000 0.500000 0.000000 )
ik = 2 ( 0.487805 0.487805 0.000000 )
ik = 3 ( 0.475610 0.475610 0.000000 )
ik = 4 ( 0.463415 0.463415 0.000000 )
ik = 5 ( 0.451220 0.451220 0.000000 )
ik = 6 ( 0.439024 0.439024 0.000000 )
...
...
...

=== energy_eigen_values ===
ik = 1 ( 0.000000 0.500000 0.500000 ) (f)
      -0.0484324576 -0.0484324576 0.1258094928 0.1258094928 (g)
      0.2619554301 0.2619554301 0.6015285208 0.6015285208
=== energy_eigen_values ===
ik = 2 ( 0.000000 0.490000 0.490000 )
      -0.0540717201 -0.0427149632 0.1258687739 0.1258687739
      0.2607026807 0.2633829927 0.6006243932 0.6006243932
      .....
      .....
      .....

```

各項目を説明します。

- (a) k 点数。この例では141 個です。
- (b) バンド数。この例では8 です。
- (c) スピン自由度。1 か2 の値をとります。この例では1 であり、スピン分極を考慮しない計算に対応します。
- (d) フェルミエネルギーの値。半導体/絶縁体の場合価電子帯の上端のエネルギーが記述されます。単位はハートリーです。
- (e) 計算した k 点
- (f) 固有値の情報が記述されます。まずこの行で、どの k 点に対応する固有値データかが分かります。この例では、1 番目の k で、その座標は逆格子ベクトルを基底として(0,0.5,0.5) となります。
- (g) 固有値のデータが、バンドの数だけ出力されます。単位はハートリーです。

スピンを考慮した計算の場合(上記の(c) が2 の場合) もほぼ同様のファイル形式ですが、上記の(e) の隣に“UP” か “DOWN” と記述される、という違いがあります。それぞれ多数派スピンと少数派スピンに対応する固有値が書き出されます。

```

      .....
      .....
      .....

=== energy_eigen_values ===
ik = 1 ( 0.000000 0.000000 0.000000) UP
      -0.1998699758 0.0267639589 0.0267639589 0.0267639589
      0.0725171077 0.0725171077 1.0289118953 1.0289118953
      1.0289118953 1.1650173104 1.1650173104 1.1650173104
      1.2129026022 1.2129026022 1.2994754011 1.2994754011
      1.2994754011 1.6365336765 2.2629596795 2.2629596795
=== energy_eigen_values ===
ik = 2 ( 0.000000 0.000000 0.000000) DOWN
      -0.1960420390 0.1062941746 0.1062941746 0.1062941746

```

0.1799862148	0.1799862148	1.0183970612	1.0183970612
1.0183970612	1.2174266166	1.2174266166	1.2192701193
1.2192701193	1.2192701193	1.3289165100	1.3289165100
1.3289165100	1.6910264603	2.2876818717	2.2876818717
		
		
		

このようなデータからバンド構造図を作成するのは手間がかかりますが、**PHASE** にはこの結果からバンド構造図を簡単に作成する“band.pl”という Perl スクリプトが付属しています。band.pl は、以下のように実行します。

```
% band.pl nfenergy.data bandkpt.in -erange=-10,10 -color -with_fermi
```

例として、体心立方鉄のバンド構造図を図 4-14 に示します。

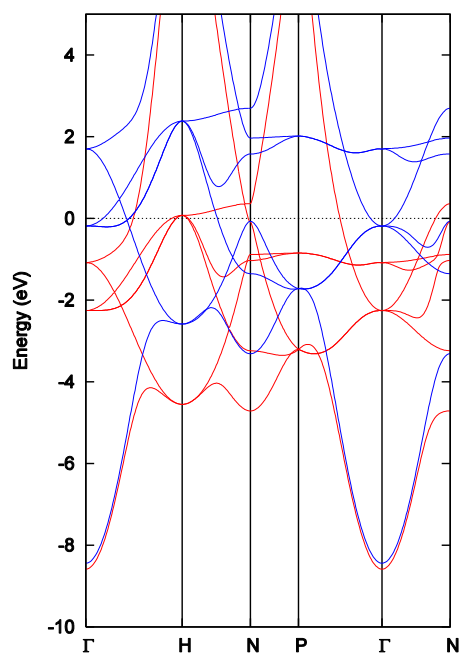


図 4-14 体心立方鉄のバンド構造図。

4.10 格子定数

4.10.1 計算方法

格子定数は、複数の格子定数において全エネルギーを計算することによって計算することが可能です。特に、立方晶の場合は次のマーナハンの状態方程式にフィットすることによって 格子定数だけではなく体積弾性率ももとめることが可能です。

$$E_{\text{tot}}(V) = \frac{BV}{B' (B' - 1)} \left[B' \left(1 - \frac{V_0}{V} \right) + \left(\frac{V_0}{V} \right)^{B'} - 1 \right] + E_{\text{tot}}(V_0)$$

ここで $E_{\text{tot}}(V)$ は単位胞の体積 V における全エネルギー、 B は体積弾性率、 B' は体積弾性率の体積微分、 V_0 は安定な格子定数における単位胞の体積です。 $B, B', V_0, E_{\text{tot}}(V_0)$ の4つがフィッティングパラメータです。

4.10.2 計算例 : Si 結晶

Si 結晶の格子定数の計算例です。この例題は、`samples/Si_lat` です。 `Si_lat` の下には、さらに `volxxx` というサブディレクトリが存在します。各々のサブディレクトリは、`xxx` という単位胞の体積に対応した入力データが格納されています。たとえば、`vol1200` というディレクトリにおける計算モデルの指定は以下のようになっています。

```
structure{
  element_list{
    #tag      element      atomicnumber
          Si      14
  }
  atom_list{
    atoms{
      #units angstrom
      #tag      element      rx      ry      rz
          Si      0.125  0.125  0.125
          Si     -0.125 -0.125 -0.125
    }
    coordinate_system = internal
  }
  unit_cell{
    a_vector = 10.62658569182611066038 0 0
    b_vector = 0 10.62658569182611066038 0
    c_vector = 0 0 10.62658569182611066038
  }
  symmetry{
    method = automatic
    tspace{
      lattice_system = facecentered
    }
    sw_inversion = on
  }
  unit_cell_type = bravais
}
```

座標データは、フラクショナルな座標データで指定しています。カルテシアンでもよいのですが、格子定数を変えるたびに座標値も変えるのは手間がかかるので、格子定数の計算にはフラクショナル座標が適していると言えます。

unit_cell_type として bravais を採用し、さらに lattice_system に facecentered を指定しています。このようにすることによって、格子定数の指定がしやすいブラベー格子によって入力の格子を指定し、実際の計算はより負荷の少ない基本格子で行うことが可能となります。

実際の計算はブラベー格子ではなく基本格子で行われるので、体積としてブラベー格子の値を採用するのであれば必要に応じて結果を変換する必要があります。たとえば、この例の場合体積弾性率は 4 倍します（面心立方格子のブラベー格子の体積は基本格子の 4 倍のため）。

計算を行い、マーナハンの状態方程式にフィットした結果を図 4-15 と表 4.4 に示します。別途計算した原子の全エネルギーから、凝集エネルギーも示しています。原子あたりの凝集エネルギーは、原子の全エネルギーから最安定の格子定数における結晶の全エネルギーを原子数で割った値を引くことによって得ることができます。

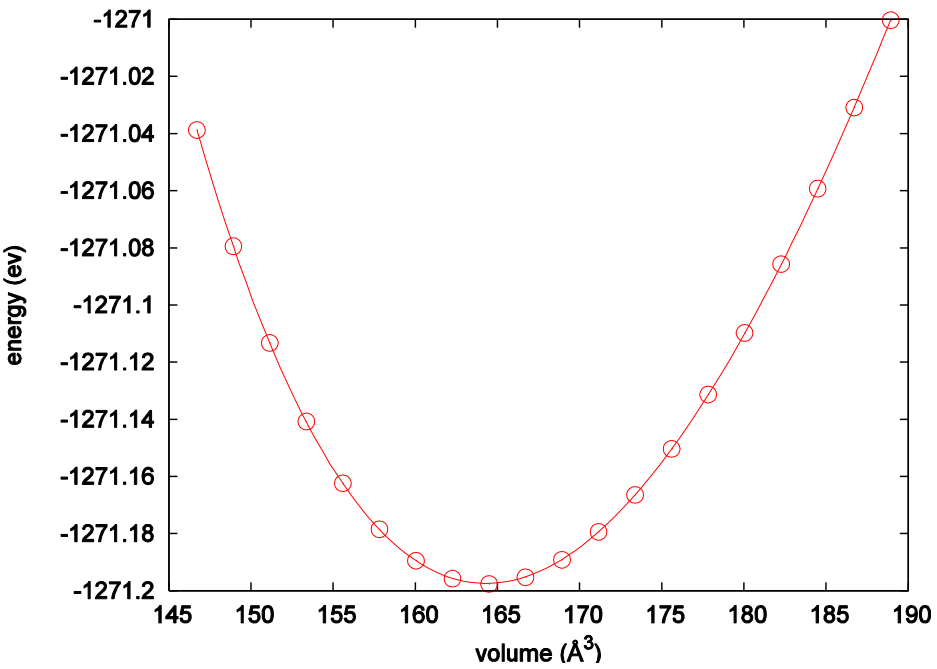


図 4-15 シリコンの Energy-Volume 曲線。白丸は計算値，実線はフィットした結果。

表 4.4 得られた格子定数と体積弾性率

	PHASE	実測データ
a (Å)	5.48	5.43
B (GPa)	87.5	98.8
Ecoh (eV/atom)	4.60	4.63

5. 応用機能

5.1 解析機能

5.1.1 ストレステンソル

5.1.1.1 機能の概要

PHASE には、ストレステンソルを計算する機能があります。ストレステンソルを計算することにより、安定な格子定数や弾性定数を計算することができます。

5.1.1.2 入力パラメータ

ストレステンソルを計算するには、入力パラメータファイル `nfnp.data` において、`structure_evolution` ブロックの下に `stress` ブロックで、ストレステンソル計算を有効にする指定をします。

Si（立方晶）の入力パラメータファイルの例を以下に示します。計算例題は、`samples/stress/`です。

```
Control{
  condition = initial
  cpumax = 24 hour
}
accuracy{
  cutoff_wf = 20.25 rydberg
  cutoff_cd = 81.00 rydberg
  num_bands = 20
  ksampling{
    mesh{ nx = 8, ny = 8, nz = 8 }
  }
}
structure{
  unit_cell_type = primitive
  unit_cell{
    #units angstrom
    a_vector = 0.000000 2.729685 2.729685
    b_vector = 2.729685 0.000000 2.729685
    c_vector = 2.729685 2.729685 0.000000
  }
  symmetry{
    method = automatic
  }
  atom_list{
    coordinate_system = internal
    atoms{
      #tag    rx    ry    rz    element  mobile  weight
      0.125  0.125  0.125    Si    yes      1
      -0.125 -0.125 -0.125    Si    yes      1
    }
  }
  element_list{
    #tag    element  atomicnumber
    Si      14
  }
}
structure_evolution{
  stress{
```

```

        sw_stress=on
    }
}

```

SCF 計算と同様に PHASE を実行します。

```
% mpirun ~/phase0_2019.01/bin/phase
```

計算が終了したら結果を確認します。

```
% grep -A3 'Total STRESS TENSOR' output000
```

```

STRESS TENSOR
  0.00000004032      0.00000000000      0.00000000000
  0.00000000000      0.0000004032      0.00000000000
  0.00000000000      0.00000000000      0.0000004032

```

ストレストンソルは

$$\begin{pmatrix} X_x & X_y & X_z \\ Y_x & Y_y & Y_z \\ Z_x & Z_y & Z_z \end{pmatrix}$$

の形式で出力されています。出力されている値の単位は [Hartree/Bohr³] です。上の結果では入力データとしてわずかに格子定数を小さく取っているため、正の X_x, Y_y, Z_z が出力されています。

また、釣り合いの位置からの格子変形($\equiv e$)、スティフネス定数($\equiv c$)を用いると 次のようなフックの法則が成り立ちます。

$$\left. \begin{aligned} X_x &= c_{11}e_{xx} + c_{12}e_{yy} + c_{12}e_{zz} \\ Y_y &= c_{12}e_{xx} + c_{11}e_{yy} + c_{12}e_{zz} \\ Z_z &= c_{12}e_{xx} + c_{12}e_{yy} + c_{11}e_{zz} \\ X_y(=Y_x) &= c_{44}e_{xy} \\ Y_z(=Z_y) &= c_{44}e_{yz} \\ Z_x(=X_z) &= c_{44}e_{zx} \end{aligned} \right\}$$

5.1.1.3 弾性定数

ストレストンソルの計算結果から、弾性定数の計算を行う例を紹介します。弾性定数は、歪みのない結晶と歪みのある結晶のストレストンソルを利用すると、上述のフックの法則より計算することができます。ここでは、Si (立方晶) を例に説明します。この例題の入力ファイルは、samples/elastic/Si の s0 および sxx の下にあります。s0 が歪みのない結晶の入力データ、sxx が xx 方向に歪みを与える場合の入力データです。

弾性定数を計算する場合、ストレストンソルの絶対値がなるべく小さくなる格子定数を利用することが望ましいです。この例題は、ストレストンソルの各成分の絶対値が 10⁻⁹ [Hartree/Bohr³] 以下となる、以下のような格子定数を採用しています。

```

unit_cell{
    #units angstrom
    a_vector = 0.000000 2.723515 2.723515
    b_vector = 2.723515 0.000000 2.723515
    c_vector = 2.723515 2.723515 0.000000
}

```

ディレクトリー samples/elastic/Si/s0 においてストレストンソルを計算すると以下のような結果が出力されます。

```
% grep -A3 'Total STRESS TENSOR' output000
```

```

Total STRESS TENSOR
  0.00000000006      0.00000000000      0.00000000000
  0.00000000000      0.00000000006      0.00000000000

```

0.00000000000	0.00000000000	0.00000000006
---------------	---------------	---------------

つぎに、ディレクトリ **sxx** へ移ります。

```
% cd ../sxx
```

このディレクトリーに置かれている入力ファイルは **s0** のものとはほぼ同じですが、以下の設定によって 11 方向(**xx** 方向)に **0.001** という格子歪みを与えた計算を行うことになります。

```
structure{
  ...
  ...
  strain{
    sw_strained_cell = on
    e11 = 0.001
  }
}
```

structure ブロックの下に **strain** ブロックを作成し、変数 **sw_strained_cell** を **on** とすると格子歪みを与えて計算を行うことができます。さらに変数 **e11, e12, e13, e22, e23, e33** で非ゼロとする歪み成分を指定します。このディレクトリで計算を行うと、以下のようなストレステンソルが得られます。

```
% grep -A3 'Total STRESS TENSOR' output000
```

Total STRESS TENSOR		
-0.0000052984	0.00000000000	0.00000000000
0.00000000000	-0.0000019980	0.00000000000
0.00000000000	0.00000000000	-0.0000019980

この例題の弾性定数の計算には、ストレステンソルの対角成分を用います。この計算例題では、回転や剪断ひずみを与えていませんので、非対角項は **0** になっています。

得られたストレステンソルから、スティフネス定数 c_{11} , c_{12} を次式から計算します。

$$\left. \begin{aligned} X_x &= c_{11}e_{xx} + c_{12}e_{yy} + c_{12}e_{zz} \\ Y_y &= c_{12}e_{xx} + c_{11}e_{yy} + c_{12}e_{zz} \\ Z_z &= c_{12}e_{xx} + c_{12}e_{yy} + c_{11}e_{zz} \\ X_y(=Y_x) &= c_{44}e_{xy} \\ Y_z(=Z_y) &= c_{44}e_{yz} \\ Z_x(=X_z) &= c_{44}e_{zx} \end{aligned} \right\}$$

今のケースでは e_{xx} 以外は **0** なので、歪みのない系とある系の X_x, Y_y の差を歪み量 (0.001) で除すれば c_{11} と c_{12} を計算することができます。結果は下記の通りです。

$$c_{11} = 5.30 \times 10^{-6} \text{ Hartree/Bohr}^3 = 156 \text{ GPa}$$

$$c_{12} = 2.00 \times 10^{-6} \text{ Hartree/Bohr}^3 = 58.7 \text{ GPa}$$

一方、弾性定数(ヤング率($\equiv Y$)・ポアソン比($\equiv P$)・体積弾性率($\equiv B$))はスティフネス定数を用いて次のような式で書き表されます。剛性率は $Y/(2 + 2P)$ と書けます。

$$\left. \begin{aligned} Y &= \frac{c_{11}^2 + c_{11}c_{12} - 2c_{12}^2}{c_{11} + c_{12}} \\ P &= \left| \frac{c_{12}}{c_{11} + c_{12}} \right| \\ B &= \frac{c_{11} + 2c_{12}}{3} \end{aligned} \right\}$$

これにスティフネス定数 c_{11} , c_{12} を代入すれば Si の弾性定数が以下のように求まります。

$$Y = 124 \text{ GPa}$$

$$P = 0.274$$

$$B = 91 \text{ GPa}$$

より精度の高い弾性定数の計算を行ないたい場合、`cutoff_wf`, `cutoff_cd` を大きめにとり、電子状態を十分に収束させる必要があり、計算時間のかかる計算になります。

5.1.1.4 ストレステンソルの補正

PHASE/0 によるストレステンソルの計算は、精度が低い場合があります。原因は、“格子がひずむことによる平面波数の変化の効果”がとりいれられていないからです。この効果を取り入れることによって、ある程度補正を行うことが可能です。

(1) 方法1.

運動エネルギーの計算における G ベクトルの高周波成分をスミアすることによって、“平面波数一定”の状況を“カットオフエネルギー一定”の状況に近づけることができます。文献(J. Phys. Chem. Solids 56 501 (1995))では、運動エネルギーの高周波成分を以下のように置き換えることが提案されています。

$$G^2 \rightarrow G^2 + A \left[1 + \operatorname{erf} \left(\frac{1/2 G^2 - E_0}{\sigma} \right) \right].$$

PHASE/0 では、上式を利用したストレステンソルの計算を行うことができます。以下のような設定を入力パラメーターファイルに記述します。

```
structure_evolution{
  lattice{
    sw_optimize_lattice = on
  }
  stress{
    sw_smear_KE = on
    a = 15 rydberg
    sigma = 0.1 rydberg
    e0 = 35 rydberg
  }
}
```

structure_evolution の下に stress ブロックを作成し、設定を行います。sw_smear_KE=on とするとこの機能が有効になります。a, sigma, e0 には対応するパラメーターを指定します。

デフォルト値は a=0.375, ecut, sigma = 0.1 Rydberg, e0=ecut-1 Rydberg です。

(2) 方法2. (バージョン 2019.01 以上)

複数のカットオフエネルギーによる計算から誤差を見積もることができます。ターゲットカットオフエネルギーを E_c 変化量を ΔE_c 全エネルギーの変化量を ΔE_t とすると、ストレスの誤差 σ_e は以下のように見積もることができます。

$$\sigma_e = -(2E_c / 3V) \times (\Delta E_t / \Delta E_c).$$

この補正を PHASE/0 に計算させるには、以下のように stress ブロックにおいて `sw_stress_correction` を on とします。

```
structure_evolution{
  stress{
    sw_stress = on
    sw_stress_correction = on
  }
}
```

```
}
}
```

ストレステンソルの補正は、カットオフエネルギーを変化させてストレステンソルを求めることによって計算します。どの程度カットオフエネルギーを変化させるかは `delta_ecut` によって指定します。カットオフエネルギーを `ecut-delta_ecut` としたケースと `ecut+delta_ecut` としたケース、そして `ecut` そのもののケースのストレステンソル計算が行われ、その後補正が計算されます。なお、補正の計算前に計算が終了した場合継続できないので注意が必要です。補正値は以下のように `output000` ファイルに記録されます。

```
!** Pulay stress :   -0.000194696412156
```

補正が計算されたあと、入力パラメーターファイルに格子最適化の設定が行われている場合補正を組み込んだ状態で格子最適化計算が始まります。そうでない場合、以下の要領で補正の値を入力ファイルに書き込み、格子最適化などを行う設定にしたうえで再度計算を実行してください（補正が必要なのは対角要素のみ、また誤差が -0.0001au だったとして）。

```
structure_evolution{
  lattice{
    sw_optimize_lattice = on
    external_stress{
      s11 = -0.0001
      s22 = -0.0001
      s33 = -0.0001
    }
  }
}
```

※ バージョン 2019.01 未満の場合補正は手動で計算する必要があります

（3）検証

これらの補正を利用し、 TiO_2 の格子定数を計算した結果を以下の表にまとめました。方法 1. のパラメーターはデフォルト値、方法 2. の ΔE_c は $\pm 5\text{Rydberg}$ としました。

	a (bohr)	c (bohr)
カットオフ 36Rydberg, EV 曲線	8.8017	5.6355
カットオフ 36Rydberg, 補正なし	8.6825	5.5862
カットオフ 36Rydberg, 方法 1.	8.7593	5.6072
カットオフ 36Rydberg, 方法 2.	8.8052	5.6200
カットオフ 80Rydberg, 補正なし	8.7918	5.6158

方法 1. 2. とともに改善しています。特に、方法 2. を使うと EV 曲線から求めた格子定数とほぼ同じ格子定数が得られています。

5.1.2 局所状態密度と部分電荷密度

5.1.2.1 機能の概要

計算した電子状態を解析するため状態密度や電子密度を描きますが、複雑な物質になると解析が困難になります。原子領域の状態密度を求めることにより、結合状態の解析が可能となります。積層構造や界面構造の場合に層毎の状態密度を計算すると、積層による電子状態の変化の解析や界面状態の同定ができます。固有エネルギーがあるエネルギー範囲に収まる電子状態からなる部分電荷密度を計算すると、それらの電子状態の分布が明瞭に分かります。原子分割と層分割の局所状態密度と部分電荷密度の計算の仕方を BaO/Si(001)界面を例に説明します。簡単のため、BaO の格子定数に Si と同じ格子定数(5.43Å)を用います。そして、図 5-1 に示すように、BaO/Si(001)界面の原子構造は 5 層からなるシリコン層と 6 層からなる BaO 層を 0 で繋げた構造にします。この BaO/Si(001)界面の計算例題は samples/BaO_Si001 です。

入力パラメータの構造に関する部分は次のようになっています。

```
structure{
    unit_cell_type=bravais
    unit_cell{
        !! a_Si=5.43 A, c-axis=5*a_Si
        !! (c.f. a_BaO=5.52 A)
        !#units angstrom degree
        a = 3.83958982184, b= 3.83958982184, c= 27.15
        alpha=90.0, beta=90.0, gamma=90.0
    }

    symmetry{
        tspace{
            system = primitive
            generators {
                !#tag rotation tx ty tz
                E          0 0 0
                C2z        0 0 0
            }
        }
        sw_inversion = off
    }
    magnetic_state = para !{para|af|ferro}

    atom_list{
        coordinate_system = internal ! {cartesian|internal}
        atoms{
            !#default mobile=no
            !#tag element rx ry rz num_layer
            Ba          0.0000 0.5000 0.05 1
            O            0.5000 0.0000 0.05 1
            Ba          0.5000 0.0000 0.15 2
            O            0.0000 0.5000 0.15 2
            Ba          0.0000 0.5000 0.25 3
            O            0.5000 0.0000 0.25 3
            O            0.0000 0.5000 0.35 4
            Si           0.0000 0.0000 0.40 5
            Si           0.5000 0.0000 0.45 6
            Si           0.5000 0.5000 0.50 7
            Si           0.0000 0.5000 0.55 8
            Si           0.0000 0.0000 0.60 9
            O            0.5000 0.0000 0.65 10
            Ba           0.5000 0.0000 0.75 11
        }
    }
}
```



```

O      0.0000 0.5000 0.75 11
Ba     0.0000 0.5000 0.85 12
O      0.5000 0.0000 0.85 12
Ba     0.5000 0.0000 0.95 13
O      0.0000 0.5000 0.95 13
    }
}
element_list{ !#tag element  atomicnumber  zeta  dev
               Si             14    0.00   1.5
               Ba             56    0.00   1.5
               O               8    0.00   1.5
    }
}

```

原子構造の緩和には時間がかかるので、**mobile** を **no** に設定して構造緩和は行わないことにします。

5.1.2.2 原子分割局所状態密度

原子分割の局所状態密度を計算するにはタグ **Postprocessing** の中にタグ **dos** とタグ **ldos** を書きます。そして、タグ **dos** の中の変数 **sw_dos** を ON にし、タグ **ldos** の中の変数 **sw_aldos** を ON にします。

```

Postprocessing{
  dos{
    sw_dos = ON
    method = g
  }
  ldos{
    sw_aldos = ON
    aldost{
      crtdst = 6.0 bohr
      naldos_from = 1
      naldos_to   = 19
    }
  }
}

```

タグ **aldos** の中の変数 **crtdst** は単位格子を原子ごとにボロノイ多面体分割するときの臨界距離です。どの原子からもこの臨界距離以上離れている領域は真空領域とみなされます。真空領域の状態密度は、(原子の個数+1) 番目の原子局所状態密度として表されます。**naldos_from** と **naldos_to** に原子分割局所状態密度を計算する最初の原子と最後の原子を指定します。これを指定しないと全原子について原子分割局所状態密度が計算されます。また、タグ **atoms** の中で変数 **aldos** を **off** にした原子の局所状態密度は計算されません。変数 **aldos** よりも **naldos_from** と **naldos_to** の方が優先されます。

計算結果は **dos.data** に出力されます。状態密度図を作成するには、付属の Perl スクリプト **dos.pl** を使います。以下のようにすれば、**dos_a001.eps**, **dos_a002.eps**, ..., **dos_axxx.eps** といったポストスクリプトファイルが作成されます。

```
% ../../../../tools/bin/dos.pl dos.data -erange=-30,5 -dosrange=0,12 -mode=atom
```

BaO/Si(001)界面の原子分割局所状態密度を計算した結果を図 5-1 に示します。Si, Ba, O の原子分割局所状態密度にそれぞれの原子の特徴を見ることができます。

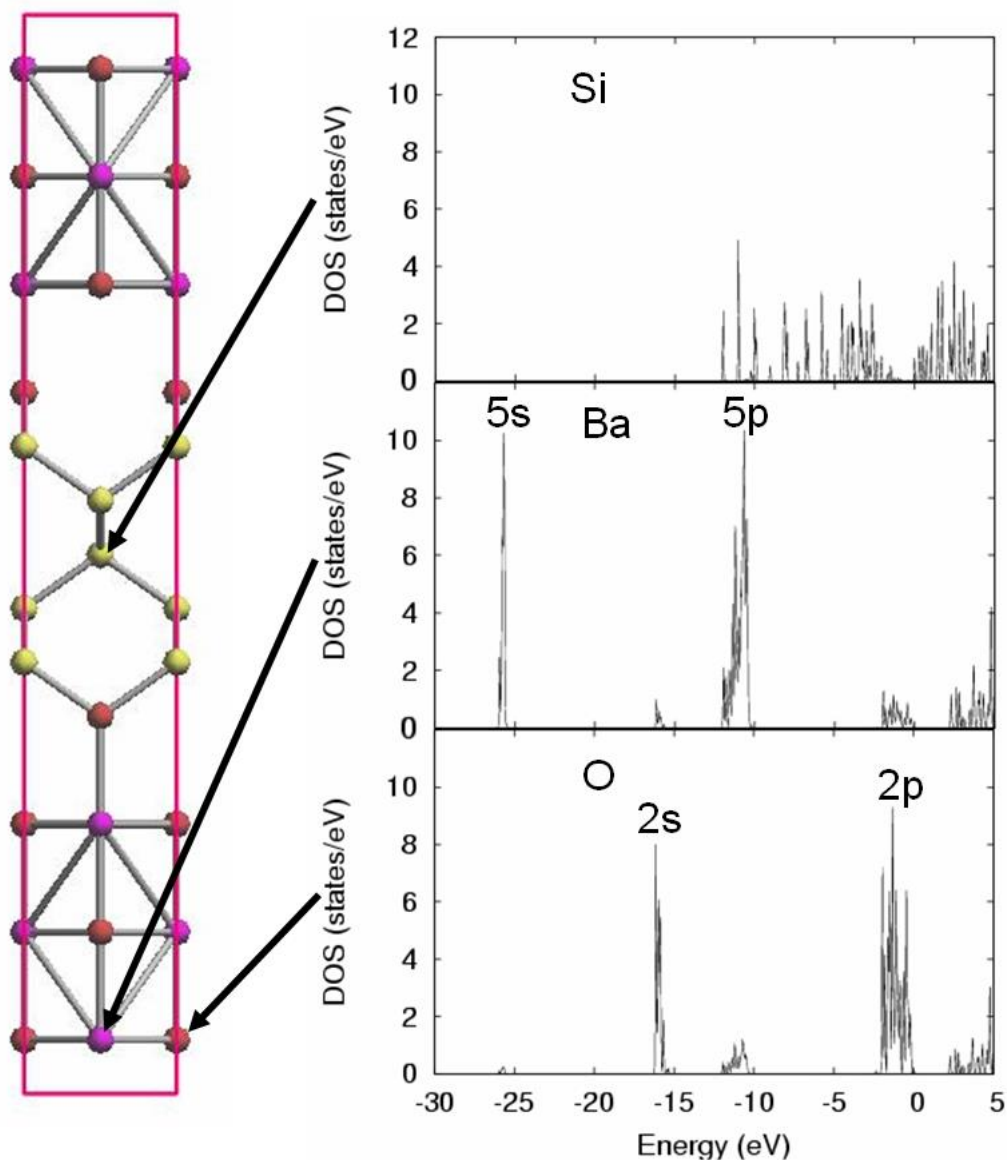


図 5-1 BaO/Si(001)界面構造の原子分割の局所状態密度。上のパネル：Si 層中央の Si の局所状態密度。中央のパネル：BaO 層中央の Ba の局所状態密度。下のパネル：BaO 層中央の O の局所状態密度。

5.1.2.3 層分割局所状態密度

層分割の局所状態密度を計算するにはタグ Postprocessing の中にタグ dos とタグ ldos を書きます。そして、タグ dos の中の変数 sw_dos を ON にし、タグ ldos の中の変数 sw_layerdos を ON にします。

```
dos{
  sw_dos = ON
  method = g
}
ldos{
  sw_layerdos = ON
  layerdos{
    slicing_way = by_atomic_positions !{regular_intervals|by_atomic_positions
  }

  deltaz = 1.0 angstrom
  normal_axis = 3
  crtdst = 3.5 bohr
```

```
}
}
```

タグ `layerdos` の中の変数 `normal_axis` では層分割するときの層の法線方向を指定します。1 が a 軸で、2 が b 軸で、3 が c 軸を表します。変数 `slicing_way` に `by_atomic_positions` を指定すると、原子位置によって局所状態密度を計算する層を定めることができます。この場合、`atoms` テーブルの `num_layer` によって、原子が含まれる層の番号を指定します。先に示した、構造の入力部分では 13 個の層に各原子を割り当てています。変数 `slicing_way` に `regular_intervals` を指定すると、ある領域を等間隔に分割して作成した各層について局所状態密度を計算します。その間隔は変数 `deltaz` に入力します。変数 `crtddst` は層を作成する領域を決める臨界距離です。端の原子からこの臨界距離まで層を作成します。

層の範囲に関する下記のような記述が `output000` に出力されます。

!!ldos	no,	min,	max
!!ldos	1	0.00000000	5.13060607
!!ldos	2	5.13060607	10.26121214
!!ldos	3	10.26121214	15.39181821
!!ldos	4	15.39181821	19.23977276
!!ldos	5	19.23977276	21.80507579
!!ldos	6	21.80507579	24.37037883
!!ldos	7	24.37037883	26.93568186
!!ldos	8	26.93568186	29.50098489
!!ldos	9	29.50098489	32.06628793
!!ldos	10	32.06628793	35.91424248
!!ldos	11	35.91424248	41.04484855
!!ldos	12	41.04484855	46.17545462
!!ldos	13	46.17545462	51.30606069
!!ldos	14	0.00000000	0.00000000

`no` は層の番号です。`min` と `max` は層の下端の位置と上端の位置を示します。最後の層は指定した以外の領域です。

計算結果は `dos.data` に出力されます。状態密度図を作成するには、付属の Perl スクリプト `dos.pl` を使います。以下のように実行すると、ポストスクリプトファイル `dos_l001.eps, dos_l002.eps, ..., dos_lxxx.eps` が作成されます。

```
% ../../tools/bin/dos.pl dos.data -erange=-20,5 -dosrange=0,20 -mode=layer
```

BaO/Si(001)界面の層分割局所状態密度を計算した結果を図 5-2 に示します。

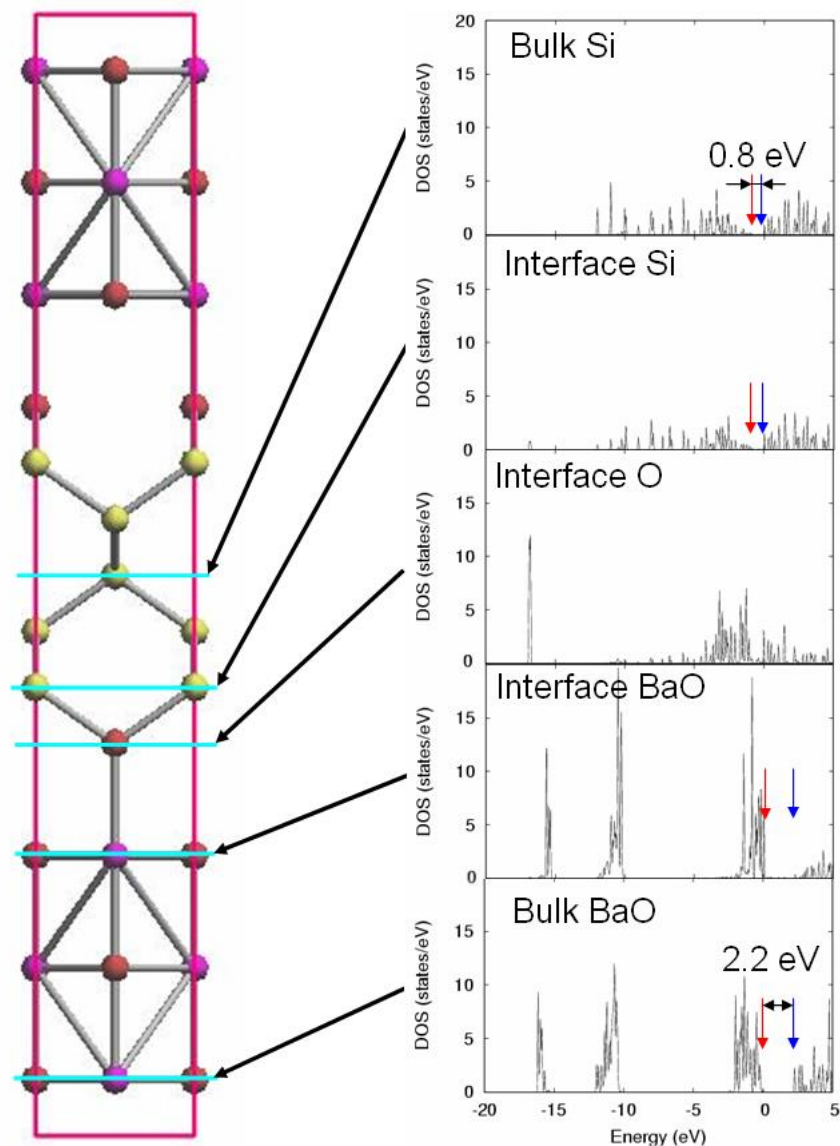


図 5-2 BaO/Si (001) 界面構造の層分割局所状態密度。一番上のパネル：Si 層の中央領域の局所状態密度。上から二番目のパネル：BaO/Si (001) 界面の Si 側の局所状態密度。中央のパネル：BaO/Si (001) 界面の酸素あたりの局所状態密度。下から二番目のパネル：BaO/Si (001) 界面の BaO 側の局所状態密度。一番下のパネル：BaO 層の中央領域の局所状態密度。

5.1.2.4 部分電荷密度

部分電荷密度を計算するにはタグ `Postprocessing` の中のタグ `charge` で指定します。タグ `charge` の中の変数 `sw_charge_rspace` とタグ `partial_charge` の中の変数 `sw_partial_charge` を `On` にします。また、この計算を行う場合には、`sw_charge_rspace` も `On` にしておく必要があります。

```
Postprocessing{
  charge{
    sw_charge_rspace = on
    partial_charge {
      sw_partial_charge = on
      Erange_min = -0.45 eV
      Erange_max = 0.45 eV
      Erange_delta = 0.05 eV
      partial_charge_filetype=separate !{individual,separate| integrated}
    }
  }
}
```

```

}
}

```

変数 **Erangle_min** と **Erangle_max** にエネルギー領域の最大値と最小値を入力します。エネルギーは金属の場合フェルミレベルから測り、絶縁体の場合は価電子帯上端のエネルギーから測ります。変数 **Erangle_delta** に入力した値の間隔のエネルギー窓を先のエネルギー領域に作成します。

出力ファイル output000 には以下の様にエネルギー窓に関する出力があります (“!pc”で始まる行)。

!pc nEwindows = 20, nvb_windows = 10, ncb_windows = 10 <<m_ESoc_set_nEwindows_pc>>									
!pc iw if_elec_state		erange(hartree)		erange(eV)					
!pc		(asis)		(shifted)		(shifted)			
!pc	1	1	(0.094537 0.096374)	(-0.018375 -0.016537)	(-0.500000 -0.450000)				
!pc	2	1	(0.096374 0.098211)	(-0.016537 -0.014700)	(-0.450000 -0.400000)				
!pc	3	1	(0.098211 0.100049)	(-0.014700 -0.012862)	(-0.400000 -0.350000)				
!pc	4	1	(0.100049 0.101886)	(-0.012862 -0.011025)	(-0.350000 -0.300000)				
!pc	5	0	(0.101886 0.103724)	(-0.011025 -0.009187)	(-0.300000 -0.250000)				
!pc	6	1	(0.103724 0.105561)	(-0.009187 -0.007350)	(-0.250000 -0.200000)				
!pc	7	1	(0.105561 0.107399)	(-0.007350 -0.005512)	(-0.200000 -0.150000)				
!pc	8	0	(0.107399 0.109236)	(-0.005512 -0.003675)	(-0.150000 -0.100000)				
!pc	9	0	(0.109236 0.111074)	(-0.003675 -0.001837)	(-0.100000 -0.050000)				
!pc	10	1	(0.111074 0.112911)	(-0.001837 0.000000)	(-0.050000 0.000000)				
!pc	11	1	(0.112911 0.114749)	(0.000000 0.001837)	(0.000000 0.050000)				
!pc	12	0	(0.114749 0.116586)	(0.001837 0.003675)	(0.050000 0.100000)				
!pc	13	0	(0.116586 0.118424)	(0.003675 0.005512)	(0.100000 0.150000)				
!pc	14	0	(0.118424 0.120261)	(0.005512 0.007350)	(0.150000 0.200000)				
!pc	15	0	(0.120261 0.122099)	(0.007350 0.009187)	(0.200000 0.250000)				
!pc	16	1	(0.122099 0.123936)	(0.009187 0.011025)	(0.250000 0.300000)				
!pc	17	1	(0.123936 0.125773)	(0.011025 0.012862)	(0.300000 0.350000)				
!pc	18	0	(0.125773 0.127611)	(0.012862 0.014700)	(0.350000 0.400000)				
!pc	19	0	(0.127611 0.129448)	(0.014700 0.016537)	(0.400000 0.450000)				
!pc	20	0	(0.129448 0.131286)	(0.016537 0.018375)	(0.450000 0.500000)				

nEwindows はエネルギー窓の総数です。nvb_windows と ncb_windows はそれぞれ価電子状態と伝導電子状態を含むエネルギー窓の数です。この設定例では、Erangle_min= -0.45 eV, Erangle_max= 0.45 eV, Erangle_delta= 0.05 eV なので、エネルギー窓の数は 18 個になるはずですが、実際には低エネルギー側および高エネルギー側にそれぞれ Erangle_delta 幅の袖領域を設けて出力するのでエネルギー窓の総数が 20 個になっていることに、ご注意ください。iw はエネルギー窓の番号です。if_elec_state はそのエネルギー窓に電子状態があるかどうかを示しています。この値が 0 の時は電子状態がなく、1 の時には電子状態が存在します。列 asis には原子単位でエネルギー窓の範囲が示されています。二つの列 shifted にはエネルギーの基準から測ったときのエネルギー窓の範囲が原子単位と eV 単位で示されています。

変数 partial_charge_filetype に individual または separate を指定すると、各エネルギー窓ごとに計算された電荷密度が番号付けされたファイルに出力されます。その際の名前の付け方は、スピン分極がない場合であれば、F_CHR = nfchr.cube に対して nfchr.00xx.cube (xx には上の表の iw の値が入る) というようになります。スピン分極がある場合には、F_CHR = nfchr.cube に対して、nfchr.up.00xx.cube, nfchr.down.00xx.cube の二種類のファイルが生成されます。上の表で if_elec_state が 0 になっているのは、その範囲に固有値がある状態がないことを示しています。その場合、cube ファイルは生成されません。

integrated を選択すると各電荷密度がひとつのファイルに追記され、各電荷密度データの先頭には PARTIALCHARGE が記述され、終わりには END が記述されます。

BaO/Si(001)界面の部分電荷密度を計算した結果を図 5-3 に示します。

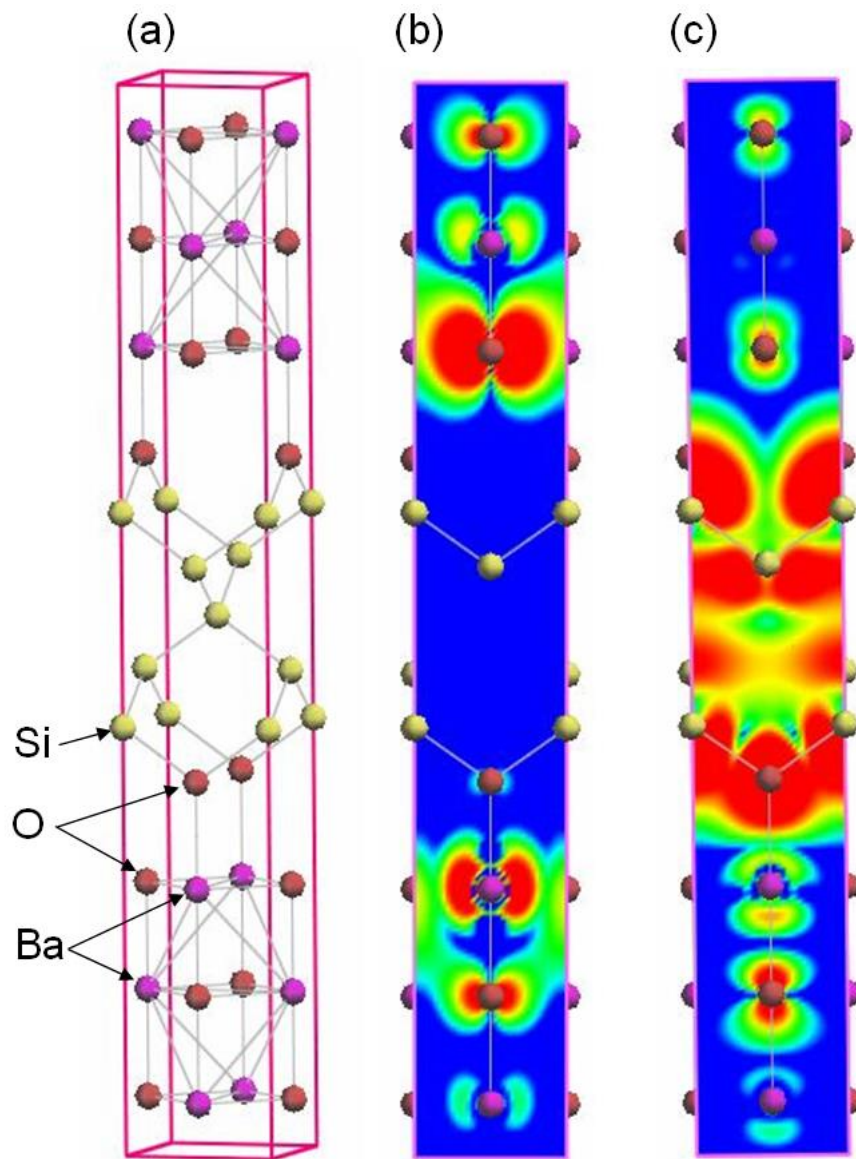


図 5-3 BaO/Si (001) 界面構造の部分電荷密度。(a) BaO/Si (001) 界面構造のモデル図。(b) フェルミレベル直下(固有エネルギーが -0.05eV から 0.0eV まで)の電子状態の部分電荷密度。(c) フェルミレベル直上(固有エネルギーが 0.0eV から 0.05eV まで)の電子状態の部分電荷密度。電子密度は 1×10^{-5} から 1×10^{-3} までが示されています。青い部分には電子が少なく、赤い分には電子が多くなっています。

- 応用例：STM 像の解析

部分電荷密度出力機能を利用すると、STM 像を模擬することが可能です。解析したいバイアスポテンシャルに対応したエネルギーウィンドウの部分電荷密度を、表面からある程度離れた平面に投影した像が計算上の STM 像です。以下、サンプルデータ(samples/stm_by_pcharge 以下) を利用して計算方法を具体的に説明します。サンプルは、Si の(001) 面に相当するデータです。通常の PHASE 入力に、以下のように部分電荷密度出力の設定を加えています。

```
postprocessing{
  charge{
    sw_charge_rspace = on
    filetype = cube
    partial_charge{
      sw_partial_charge = on
      partial_charge_filetype = individual
      Erange_min = 0 eV
      Erange_max = 0 eV
      Erange_delta = 1 eV
    }
  }
}
```

このように設定することによって、フェルミエネルギーからみて-1 eV から 0 eV までのデータと 0 eV から 1 eV のエネルギーウィンドウの部分電荷密度が出力されます。それぞれ、-1 V (占有状態) および 1 V (非占有状態) のバイアスポテンシャルに対応した STM 像が得られます。この入力データを利用して計算を実施すると、nfchr.0001.cube (-1 eV から 0 eV の電荷密度データファイル) と nfchr.0002.cube (0 eV から 1 eV の電荷密度データファイル) が作成されます。それぞれ、表面から 5 Å 程度離れた地点でのコンター図を占有状態について図 5-4(a)に、非占有状態について図 5-4(b)に示します。

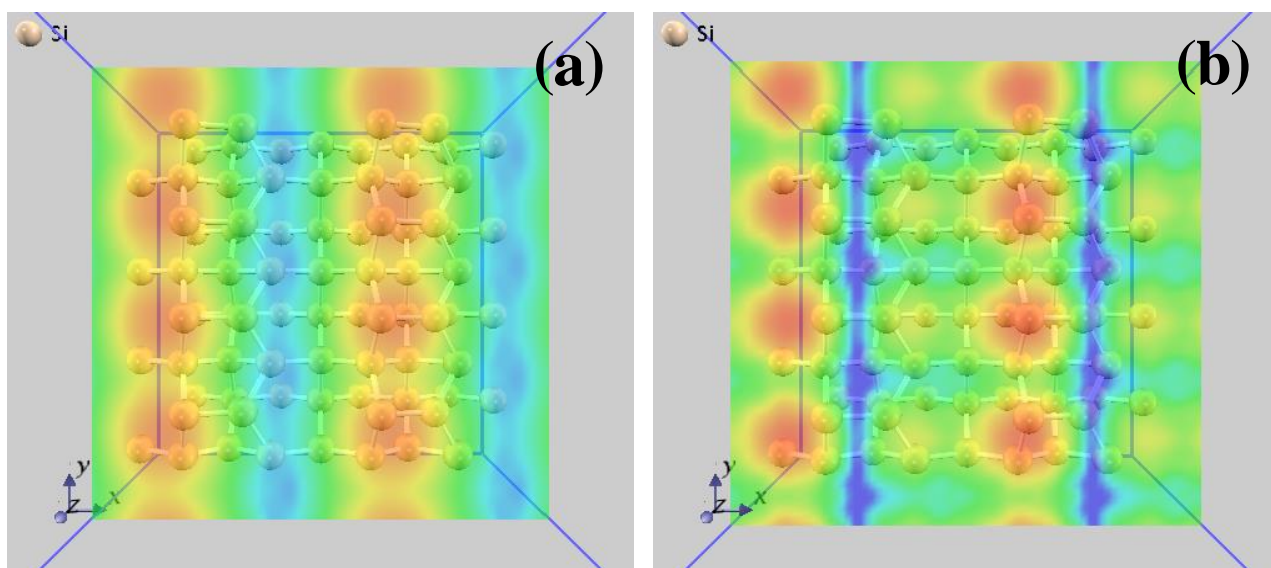


図 5-4 Si (100)面の STM 像, (a) 占有状態の像, (b)非占有状態の像。

5.1.2.5 ウルトラソフト型擬ポテンシャルを利用している場合の高速化

ウルトラソフト擬ポテンシャルを利用して局所状態密度計算する場合、非常に多くの計算時間がかかってしまう

ことがあります。これは欠損電荷の計算に時間がかかってしまうからなのですが、この計算を実空間で行うことによって高速化を実現することができます。欠損電荷の計算を実空間で行うには、以下のような設定を行います。

```
Postprocessing{
  dos{
    sw_dos = on
  }
  ldos{
    sw_rspace = on
    sw_aldos = on
    sw_layerdos = on
    aldos{
      ...
      ...
    }
    layerdos{
      ...
      ...
    }
  }
  ...
  ...
}
```

ldos ブロックで変数 `sw_rspace` を定義し、その値を `on` とすれば欠損電荷の計算を実空間で行わせることができます。

5.1.3 射影状態密度

PHASE には、軌道ごとに射影した状態密度を計算する機能もあります。ここでは、射影状態密度を計算する方法を紹介します。

5.1.3.1 入力パラメータ

射影状態密度を計算するには、射影したい軌道の設定を以下のように指定します。

```
accuracy{
  ...
  projector_list{
    projectors{
      #tag no group radius l t
      1 1 1.0 0 1
      2 1 1.0 1 1
      3 2 1.5 0 2
      4 2 1.5 1 2
      5 2 1.5 2 2
    }
  }
}
```

no に軌道の識別番号を指定します。省略可能です。**group** には、“軌道グループ”を指定します。ひとまとめに扱いたい軌道には同じ **group** 値を指定します。**radius** には軌道の半径をボーア単位で指定します。原子間距離の半分程度よりも小さな値が目安となります。デフォルト値は **1 bohr** です。**l** には、軌道角運動量を指定します。**0** が **s** 軌道、**1** が **p** 軌道、**2** が **d** 軌道、**3** が **f** 軌道に対応します。最後に、**t** に主量子数を指定します。ただし、この場合の主量子数とは擬ポテンシャルから見た場合の主量子数であり、ほとんどの場合 **1** となります。擬ポテンシャルによっては角運動量が同じ軌道が **2** つ定義されている場合があります。**2** つのうちエネルギーの高い方を指定したい場合に **t** の値を **2** としてください。

次に、定義した射影演算子を原子に割り当てます。これは、以下のように原子配置の定義において属性値 **proj_group** を追加して指定します。

```
structure{
  atom_list{
    atoms{
      #tag element rx ry rz mobile proj_group
      Fe1 0.0 0.0 0.14783 on 1
      Fe2 0.0 0.0 0.35217 on 2
      Fe1 0.0 0.0 0.85217 on 1
      Fe2 0.0 0.0 0.64783 on 2
      ...
      ...
    }
  }
}
```

磁気量子数と軌道の性格の対応表

占有行列の添え字	$l = 0$	$l = 1$	$l = 2$	$l = 3$
1	s	x	$3z^2 - r^2$	$z(5z^2 - 3r^2)$
2		y	$x^2 - y^2$	$x(5z^2 - r^2)$
3		z	xy	$y(5z^2 - r^2)$
4			yz	$z(x^2 - y^2)$

5			zx	xyz
6				$x(x^2 - 3y^2)$
7				$y(3x^2 - y^2)$

この例では、Fe1 に group が 1 の軌道グループを、Fe2 に group が 2 の軌道グループを指定しています。異種元素間では異なる軌道グループを指定する必要があります。

postprocessing ブロックにおいて射影演算子を計算するためのスイッチを有効にします。

```
postprocessing{
  ...
  pdos{
    sw_pdos = on
  }
}
```

射影状態密度の計算方法は、postprocessing の dos ブロックにおける指定に従います。

5.1.3.2 計算結果の出力

```
PDOS: ia= 2 l= 1 m= 1 t= 1
No. E(hr.) dos(hr.) E(eV) dos(eV) sum
6 -1.95781 0.0000000000 -56.762838 0.0000000000 0.0000000000
16 -1.95681 0.0000000000 -56.735626 0.0000000000 0.0000000000
26 -1.95581 0.0000000000 -56.708415 0.0000000000 0.0000000000
36 -1.95481 0.0000000000 -56.681204 0.0000000000 0.0000000000
46 -1.95381 0.0085366260 -56.653992 0.0003137151 0.0000002437
56 -1.95281 0.0176460501 -56.626781 0.0006484801 0.0000254127
```

PDOS: という文字列から始まる行が、射影状態密度データの始まりをあらわします。ia=の後に対応する原子の ID が、l=のあとに対応する軌道角運動量が、m=のあとに対応する磁気量子数が、t=のあとに対応する主量子数が出力されます。それ以降の行は、通常の状態密度データと同じです。磁気量子数と軌道の性格の対応は、表に示しています。

射影状態密度データを含んだ dos.data の処理には、dos.pl に mode=projected オプションをつけて実行します。

```
% dos.pl dos.data -mode=projected -color -with_fermi
```

実行すると、EPS 形式のファイル dos_aAAAILmMtT.eps が出力されます。AAA は原子の ID, L は軌道角運動量, M は磁気量子数, T は主量子数に対応した数字です。また、data=yes オプションを利用すると、軌道ごとに分割された状態密度データファイルを得ることができます。そのファイル名は、EPS ファイルの拡張子を data に変更したものとなります。

5.1.3.3 計算例 : BaTiO₃ 結晶の射影状態密度

BaTiO₃ 結晶の射影状態密度を計算した例です。この例題は、samples/pdos/BaTiO3 以下にあります。BaTiO₃ はペロブスカイト構造をとる結晶です。厳密には正方晶ですが、立方晶に非常に近い結晶構造です。この例では、結晶構造を以下のように指定し、立方晶として設定しています。

```
structure{
  atom_list{
    atoms{
      #units angstrom
      #tag element rx ry rz proj_group
      Ba 0.00 0.00 0.00
      O 0.50 0.50 0.00 2
      O 0.50 0.00 0.50 2
    }
  }
}
```

```

O 0.00 0.50 0.50 2
Ti 0.50 0.50 0.50 1
}
}
unit_cell{
  #units angstrom
  a_vector = 4 0.00 0.00
  b_vector = 0.00 4 0.00
  c_vector = 0.00 0.00 4
}
}

```

射影する軌道は、以下のように指定します。

```

accuracy{
  projector_list{
    projectors{
      #tag no group radius l
      1 1 1.0 2
      2 2 1.0 1
    }
  }
}

```

グループ 1 は l が 2(d 軌道)、グループ 2 は l が 1(p 軌道) であり、それぞれ Ti と O に割り当てています。最後に、**postprocessing** ブロックにおいて射影状態密度を計算する機能を有効にしています。

```

postprocessing{
  dos{
    sw_dos = on
    method = tetrahedral
  }
  pdos{
    sw_pdos = on
  }
}

```

状態密度計算は **tetrahedral** 法を利用しています。したがって、k 点サンプリングは **mesh** 法、**smearing** は **tetrahedral** 法を指定しています。

BaTiO₃ 結晶の全状態密度を図 5-5 に、Ti の d 軌道に射影した状態密度を図 5-6 に示します。

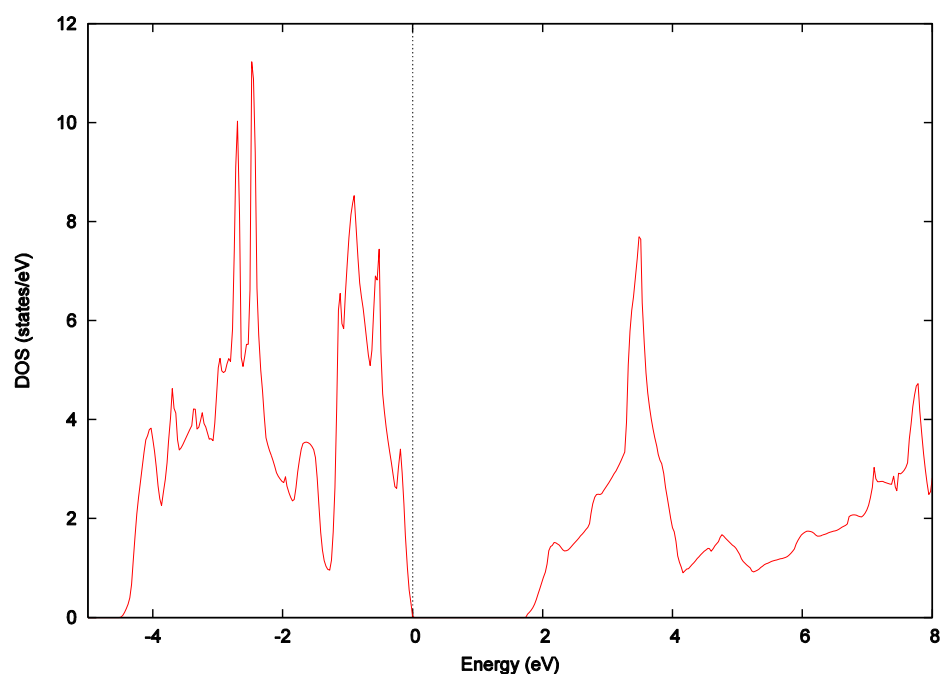


図 5-5 BaTiO3 結晶の全状態密度

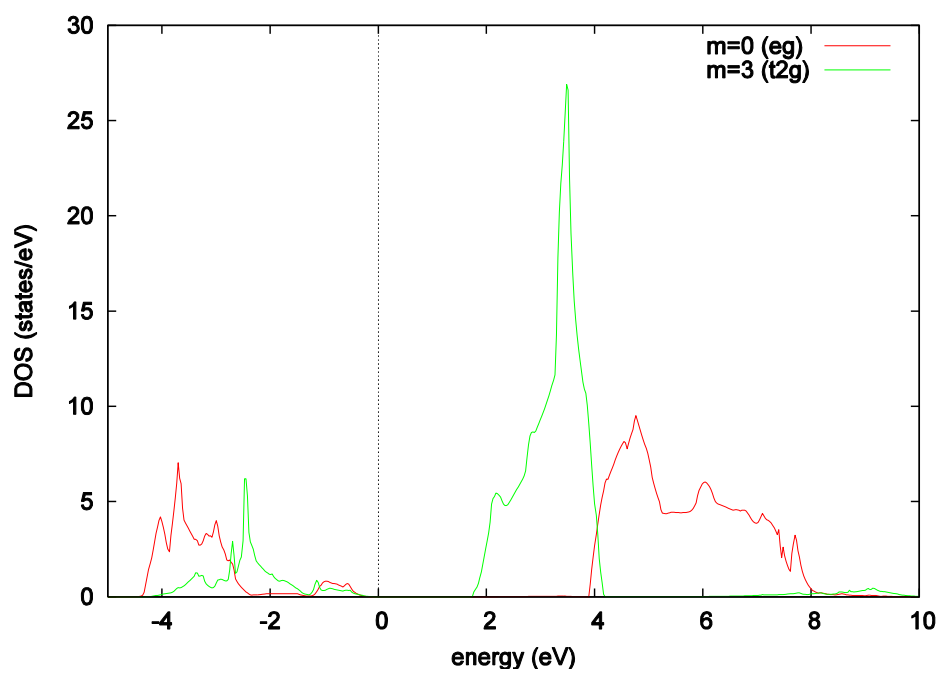


図 5-6 Ti の d 軌道の射影状態密度

5.1.4 ワニエ関数

5.1.4.1 機能の概要

電子状態は結晶全体に広がったブロッホ波と呼ばれる波として表されます。ブロッホ波を k 空間に関してフーリエ変換することにより得られる局在した関数をワニエ関数と呼びます。ワニエ関数の中心位置は電子分布の平均位置を表すため、その和から結晶の分極を容易に知ることができます。また、ワニエ関数の 2 乗は電子の分布を表すため、化学結合に関する知見が得られます。一般にワニエ関数は一意に定まりません。ワニエ関数の広がりが最小になるように変換することにより得られる、一意に定まる関数を最大局在ワニエ関数といいます。

最大局在ワニエ関数はワニエ関数の広がりを表す汎関数 (広がり汎関数) を最小にするようにブロッホ波をユニタリー変換して求めます。広がり汎関数はユニタリー変換行列の関数で、広がり汎関数をユニタリー変換行列に関して微分して得られる行列はワニエ関数の広がりを狭めるユニタリー変換の方向になっています。この方向にわずかにユニタリー変換していくことで、ワニエ関数の広がりを最小にすることができます。

5.1.4.2 計算例 : Si の最大局在ワニエ関数

最大局在ワニエ関数を計算するには、Postprocessing で Wannier 関数を計算することを指定します。

```
Postprocessing{
  wannier{
    sw_wannier = ON
    eps_grad = 1.d-3
    dt = 1.d-4
    max_iteration = 1000
    filetype = cube
  }
}
```

汎関数の勾配の大きさが `eps_grad` 以下になったら計算は終了します。`dt` は最急降下法の仮想的な時間刻みです。繰り返しが `max_iteration` を超えたら計算は停止します。ワニエ関数の出力ファイルは `Gaussian cube` 形式に指定します。ワニエ関数のファイルの拡張子が `cube` になるように、以下のように `file_names.data` にファイルポインタ `F_WANNIER` を記述します。

```
&fnames
F_INP = './nfinput.data'
F_POT(1) = './Si_ldapw91_nc_01.pp'
F_WANNIER = './nfwannier.cube'
&end
```

この機能は Γ 点のみの計算でしか使用できません。

```
ksampling{
  method = gamma
}
```

また、並列計算には対応していません。並列計算で収束した結果を得ている場合は、非並列の継続処理としてワニエ関数の計算をしてください。

Si のワニエ関数の計算例題は、`sample/wannier/Si8` です。計算を実行すると、ワニエ関数の出力として、`nfwannier.00001.cube` といったファイルが 16 個出力されます。それらの一つを可視化した結果を図 5-7 に示します。最大ワニエ関数は Si-Si 結合間に局在していることがわかります。これは、Si 結晶の結合が共有結合であることを示しています。

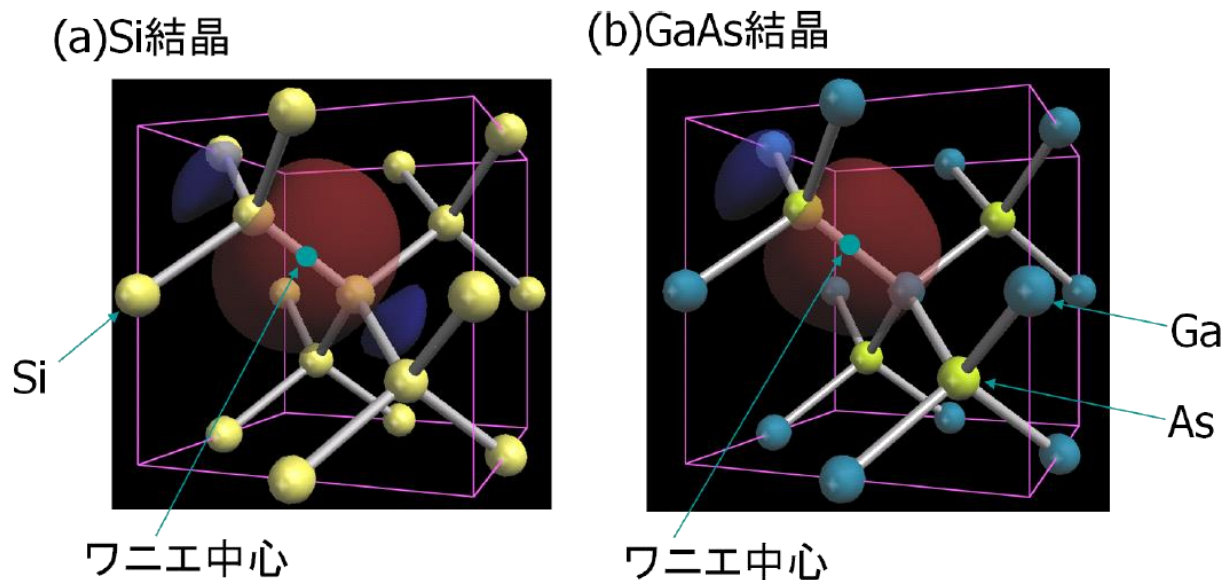


図 5-7 Si 結晶(a) と GaAs 結晶(b) の最大局在ワニエ関数

期待しない局所極小に収束してしまうことがあります。sw_random_wannier を ON にしてランダムな状態から計算を始めることでこの問題が解決できることがあります。

```
Postprocessing{
  wannier{
    ...
    sw_randomize = ON
    ...
  }
}
```

また、収束が不十分な状態で計算が終了しているようでしたら、sw_continue を ON にして計算を継続してください。

```
Postprocessing{
  wannier{
    ...
    sw_continue = ON
    ...
  }
}
```

5.1.4.3 Wannier90 を用いたワニエ関数解析

(1) 機能の概要

Wannier90 プログラム(<http://www.wannier.org/>)と連携することによってワニエ関数を出力することも可能です。Wannier90 プログラムと連携することによって、PHASE に組み込まれたワニエ関数解析機能では実施することのできない、以下の計算を行うことが可能です。

- Γ 点だけでなく、一般の k 点サンプリングでワニエ関数を出力することが可能
- ワニエ補間により、バンド構造を得ることが可能
- ワニエ補間により、フェルミ面や逆空間における等エネルギー面の描画が可能

ここでは、PHASE を Wannier90 と連携させて上記のような計算を行う方法を説明します。本機能を利用する場合は、上述のウェブサイトから Wannier90 プログラムを入手し、コンパイル作業を行ってください。

(2) 計算方法

Wannier90 と連携して解析を行うには、以下のような処理を行います。Wannier90 の入力ファイル作成方法は、Wannier90 のユーザーマニュアルを参照してください。

1. Wannier90 の入力データを作成します。ファイル名は *seedname.win* とします。*seedname* は任意の文字列で、系の名前などを指定します。
2. Wannier90 を、以下のように“プリプロセスモード”で実行します。

```
% wannier90.x -pp seedname
```

この作業によって、Wannier90 が必要とするデータの情報を記録したファイルが作成されます。このファイルの情報をもとに PHASE の入力ファイルを作成します。

3. PHASE の入力パラメータファイルを作成します。格子定数や k 点サンプリングを Wannier90 の入力に合わせて編集します。その具体例は例題において紹介します。さらに、**postprocessing** ブロックに以下を追記します。

```
postprocessing{
  wannier{
    seedname = "seedname"
    sw_wannier90 = on
    nb_wan90 = wannier90 で利用するバンド数
  }
}
```

変数 **seedname** に上述の *seedname* を二重引用符で指定します。**sw_wannier90** を **on** とすることによって Wannier90 が必要とするファイルを出力することができます。**nb_wan90** には、Wannier90 で利用するバンド数を指定します。PHASE のバンド数以下の数値にする必要があります。

4. PHASE を通常通り実行します。その結果、Wannier90 が必要とする行列要素データや固有値データの記録されたファイルが得られます。ただし、Wannier90 が利用するデータの出力は非並列計算で行う必要があるため、並列計算をしたい場合はひとまず **sw_wannier90** を **off** として計算を収束させたあとに **sw_wannier90** を **on** とし、非並列の継続計算で Wannier90 用の出力を行います。
5. Wannier90 を実行します。

```
% wannier90.x seedname
```

その結果、Wannier90 のログ出力やワニエ関数データのほか、Wannier 基底のタイトバインディングハミルトニアンデータなどが得られます。*seedname.win* の設定によっては、そのほかバンド構造やフェルミ面を構築するために必要なデータ、1次元伝導解析の結果なども得られます。

(3) ワニエ補間

`wan_interp` プログラムは、ワニエ補間を行ってバンド構造やフェルミ面の計算を行うプログラムです。そのソースコードは、`src_wan_interp` 以下にあります。コンパイルするにはこのディレクトリーへ移り、コンパイル用のシェルスクリプト、`make.sh` 中の **Fortran90** コンパイラーと **LAPACK** ライブラリーの設定を行ったあと `make.sh` を実行してください。なお、ワニエ補間によるバンド構造やフェルミ面の計算は、**Wannier90** 本体によって実施することも可能です。その方法は、**Wannier90** のユーザーマニュアルを参照してください。**Wannier90** によって行う場合は、`band.pl` によるバンド構造図の作成や **PHASE-Viewer** によるフェルミ面の描画などは行えません。

`wan_interp` プログラムは、以下のファイルを必要とします。

- `seedname_hr.dat` ファイル：**Wannier90** プログラムによって出力される、ワニエ基底のタイトバインディングハミルトニアンデータです。
- `seedname.nnkp` ファイル：**Wannier90** が必要とするデータを作成するために必要な情報が記録されたファイルです。**Wannier90** を“プリプロセスモード”で実行すると得られるファイルです。
- `kpoint.data` ファイル：計算したい k 点の情報が記録されたファイル。データフォーマットは、**PHASE** の標準の `kpoint.data` ファイルのフォーマットです。その作成方法は、バンド構造図の場合はバンド計算の項を、フェルミ面の場合は **PHASE-Viewer** ユーザーマニュアルの逆空間ビューアーの項を参照してください。
- `nfefermi.data` ファイル：フェルミエネルギーの値が記録されたファイルです。**PHASE** から出力されますので、`wan_interp` プログラムを実行するディレクトリーにコピーしておいてください。
- `fs.data` ファイル（フェルミ面の場合）：**PHASE-Viewer** がフェルミ面を描画する際に利用する補助データが記録されたファイルです。**PHASE-Viewer** の手続きによってフェルミ面用の k 点データファイルを作成した場合自動的に作成されます。

以上のファイルを揃えたら、つぎの要領で実行します。

```
% wan_interp seedname
```

(4) 計算例題

具体例として、GaAs 結晶のケースを説明します。この例題の入力ファイルは、`samples/wan90/GaAs` 以下にあります。

- ワニエ関数の出力

まず、**Wannier90** を以下のように実行します。

```
% wannier90.x -pp gaas
```

この操作によって、必要な情報が `gaas.nnkp` というファイルに出力されます。このファイルの記述をもとに **PHASE** の入力ファイルを作成します。

まず、`gaas.nnkp` ファイルには次のような書式で単位胞の情報が Å 単位で記述されます。

```
begin real_lattice
  0.0000000  2.8265001  2.8265001
  2.8265001  0.0000000  2.8265001
  2.8265001  2.8265001  0.0000000
end real_lattice
```

この指定に従い、**PHASE** の入力パラメーターファイルの単位胞の指定は以下のようになっています。

```
structure{
  unit_cell{
    #units angstrom
    a=5.653, b=5.653, c=5.653
```



```

    alpha=90, beta=90, gamma=90
  }
  symmetry{
    tspace{
      lattice_system = facecentered
    }
  }
}

```

この例では、ブラベー格子を利用して単位胞の指定を行っています。GaAs は面心立方格子なので、**lattice_system** パラメーターに **facecentered** が指定されています。

また、gaas.nnkp においては **k** 点サンプリングの方法が次のように指定されています。

```

begin kpoints
  64
  0.00000000  0.00000000  0.00000000
  0.00000000  0.00000000  0.25000000
  0.00000000  0.00000000  0.50000000
  0.00000000  0.00000000  0.75000000
  0.00000000  0.25000000  0.00000000
  0.00000000  0.25000000  0.25000000
  0.00000000  0.25000000  0.50000000
  ...
  ...
end kpoints

```

この通りに指定するため、PHASE の入力ファイルにおいては以下のように **k** 点サンプリングを“直接指定モード”を利用して指定しています。

```

accuracy{
  ksampling{
    method = directin
    kpoints{
      #tag kx ky kz denom weight
0      0    0    4    1
0      0    1    4    1
0      0    2    4    1
0      0    3    4    1
0      1    0    4    1
0      1    1    4    1
0      1    2    4    1
...
...
    }
  }
}

```

“method=directin”によってサンプルする **k** 点を直接指定することが可能です。kpoints テーブルにおいて、**kx, ky, kz, denom, weight** によって **k** 点を指定します。**k** 点の座標は(kx/denom, ky/denom, kz/denom), その重みが weight です。

Postprocessing ブロックでは、Wannier90 用の出力を行う設定が施されています。

```

postprocessing{
  wannier{

```

```

        seedname = "gaas"
        sw_wannier90 = ON
        nb_wan90 = 8
    }
}

```

Wannier90 用の出力は非並列計算で行う必要があるのですが、並列計算をしたい場合はひとまず `sw_wannier90` を `off` として計算を収束させたあとに `sw_wannier90` を `on` とし、継続計算で非並列の計算を行い、Wannier90 用の出力を行います。

PHASE を実行し、Wannier90 用のデータが得られたら、Wannier90 を `-pp` をつけずに行します。

```
% wannier90.x gaas
```

Wannier90 による計算が終了すると、`gaas_00001.xsf`, `gaas_00002.xsf`,... などのファイルが作成されますが、これは XCrysDen プログラム (<http://www.xcrysden.org/>) を利用して可視化することのできるワニエ関数データです。XCrysDen プログラムを利用すればワニエ関数を可視化することができますが、PHASE-Viewer によって可視化を行う場合は Gaussian cube 形式に変換する必要があります。この作業は、PHASE パッケージに含まれる `convpy` という Python スクリプトを利用することによって実現することができます。図 5-8 に、本例題によって得られるワニエ関数を XCrysDen で可視化した図を示します。

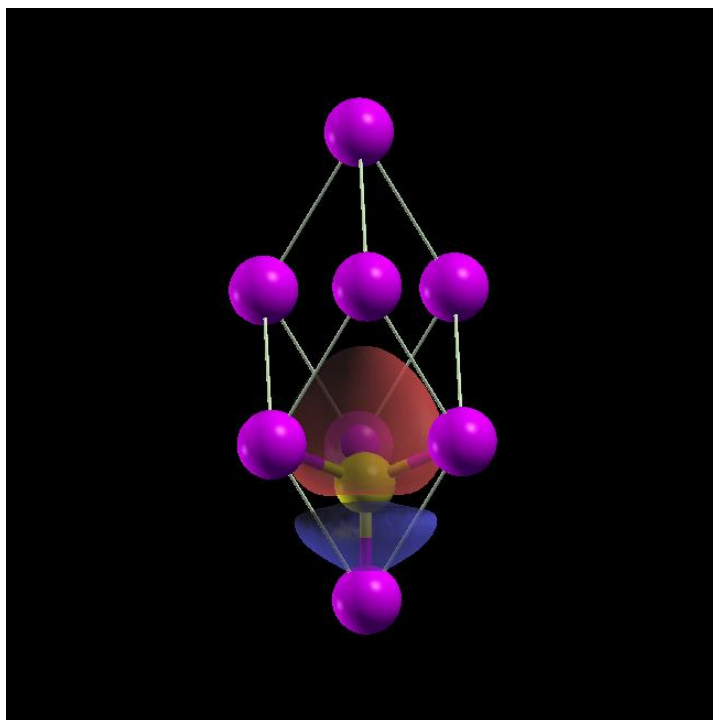


図 5-8 GaAs のワニエ関数

● ワニエ補間

Wannier90 には、計算したワニエ関数を基底にタイトバインディングハミルトニアンを構築する機能が備わっており、この機能を利用することによって通常の方法よりも少ない計算量でバンド構造やフェルミ面、等エネルギー面などの描画などを行うことが可能です。この作業は、`wan_interp` プログラムを利用して行います。この例では、GaAs の伝導体下端付近の等エネルギー面を描画する方法を紹介します。

等エネルギー面の描画に必要なデータを計算するためには、PHASE-Viewer の“逆空間ビューアー”を利用します。その方法は PHASE-Viewer ユーザーマニュアルを参照してください。“逆空間ビューアー”によって `kpoint.data` ファイルが作成されるので、このファイルと、ワニエ関数計算によって得られた `gaas.nnkp` ファイル、`gaas_hr.dat` ファイルのほか、PHASE によって得られた `nfefermi.data` ファイルを一つのディレクトリーにまとめ、そのデ

ディレクトリにおいて `wan_interp` プログラムを実行します。

```
% wan_interp gaas
```

すると、ワニエ補間によって `kpoint.data` ファイルに記録された k 点の固有値データが計算され、`nfenergy.data` というファイルに記録されます。得られたファイルを、“逆空間ビューアー”によって `kpoint.data` ファイルを作成したディレクトリにコピーします。以上の作業で等エネルギー面を描画するための準備は完了です。得られた伝導体下端付近の等エネルギー面を **PHASE-Viewer** で可視化した図を図 5-8 に示します。

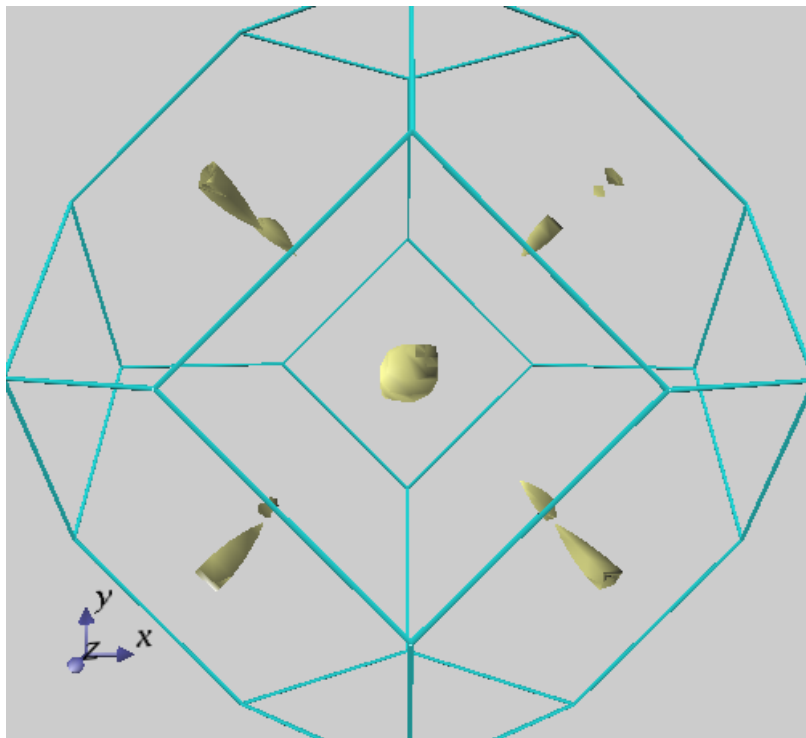


図 5-9 GaAs 結晶の伝導体下端の等エネルギー面

(5) 使用上の注意

- ワニエ関数出力は、非並列計算で行う必要があります。並列計算を行う場合はひとまず **sw_wannier90** を **off** として計算を収束させたあとに **sw_wannier90** を **on** とし、非並列の継続計算で Wannier90 用の出力を行ってください。
- 本機能で利用できる擬ポテンシャルは、ノルム保存型のみです。

5.1.5 STM 像解析

5.1.6 XPS 解析

5.1.6.1 機能の概要

X 線光電子分光(X-ray Photoemission Spectroscopy, XPS)解析の擬ポテンシャル法による第一原理計算は、内殻正孔(Core Hole)を含む原子の擬ポテンシャルを用いて、内殻準位シフト(Core Level Shift, CLS)を計算します。

(1) 内殻準位シフト

内殻準位とは原子の深い電子準位のことであり、化学結合には寄与しないくらい原子に強く局在したものです。例えばシリコン原子の場合、14 個の電子は $(1s)^2(2s)^2(2p)^6(3s)^2(3p)^2$ のように 5 個の準位を占有しますが、この中で $1s$, $2s$, $2p$ が内殻準位です。相対論的なシリコン原子の電子準位を CIAO で GGA 計算すると

Energy levels [All-electron]					
Element ---> Si					
symm	j	Energy (Ha)	Energy (eV)	nocc	focc
1s	1/2	-65.6258330748	-1785.7697073691	2	2.00000
2s	1/2	-5.1250077353	-139.4585506190	2	2.00000
2p	1/2	-3.5260321902	-95.9482139488	2	2.00000
2p	3/2	-3.5022484901	-95.3010265676	4	4.00000
3s	1/2	-0.3967820153	-10.7969875601	2	2.00000
3p	1/2	-0.1503011244	-4.0899015276	2	2.00000
3p	3/2	-0.1491437813	-4.0584086215	4	0.00000
Total number of electrons					14.00000

となり、内殻準位は化学結合の目安である数 eV より圧倒的に深いことがわかります。そのため内殻準位の波動関数は隣接した原子と重なりを持たず、エネルギー分散の無い離散準位を生じます。

シリコン原子の内殻準位を観測する手段として 100~130 eV の軟 X 線領域の単色光を照射し、放出される光電子の運動エネルギーを測定する実験方法があります[1]。ちなみに、エネルギー可変の単色光は加速器のアンジュレータから放射されたシンクロトロン放射光を用います。上記の Si 原子の電子準位を参考にすれば、この方法では $2p$ 準位から放出される光電子に着目していることがわかります。 $2p$ 準位はスピン軌道相互作用のために

$2p_{3/2}$ と $2p_{1/2}$ に0.64 eV 程度分裂します。

光電子放出の理論からよく知られているように、照射光のエネルギーを $h\nu$ とすれば放出される光電子の運動エネルギー E_{kin} は

$$E_{kin} = h\nu - W - (E_F - E_c) \quad (1)$$

となります。ここで、 W は仕事関数、 E_F は Fermi 準位、 E_c は内殻準位です。一般に結晶表面では電子のしみ出しによる電気二重層が形成されるため、内殻電子の感じるポテンシャルは表面から外側になればなるほど浅くなります。そのため、内殻準位も表面では浅くなります。この関係を模式的に示したのが図 5-10 です。その他、化学結合にともない原子のポテンシャルが上下するので、内殻準位はこれに連動した効果も受ける。この 2 つの効果のため内殻エネルギー準位 E_c は原子により異なる値をとります。内殻準位シフトの実験では、表面から十分内部に入ったバルク位置の原子による内殻準位を基準にして表面付近の化学結合が異なる原子の内殻準位のエネルギーの差を測定することにより、表面付近の原子の化学結合状態や構造を推定します。

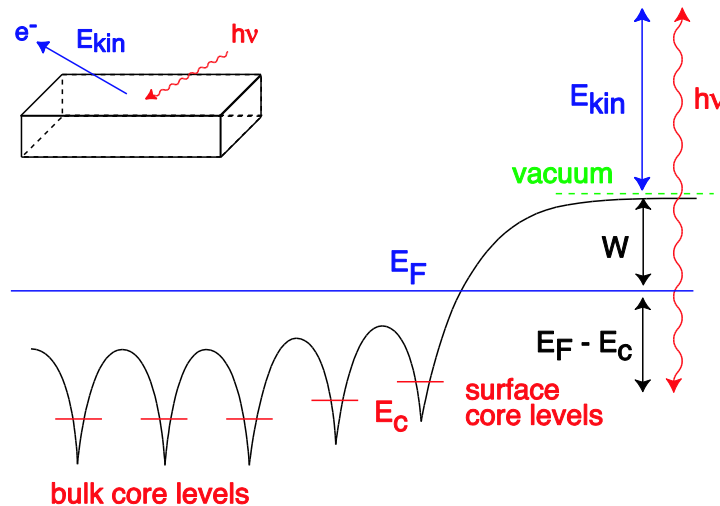


図 5-10 光電子放出過程のエネルギープロファイル：図の右側が結晶表面，左側がバルクである。Fermi 準位 E_F は一定であるのに対して，内殻準位 E_c は原子により異なり，表面付近ではバルクにくらべて内殻準位が浅くなる。

さて，内殻準位シフトを第一原理計算するためには，式(1)によれば 個々の原子の内殻準位 E_c を計算します。このエネルギー値 E_c は内殻電子も扱う全電子計算を行えば直ちに得られますが，価電子のみを扱う擬ポテンシャル法からは得られません。この理由から，擬ポテンシャル法では計算できないと思われていますが，この問題を見事に解決したのが Scheffler のグループ[2]です。Scheffler らは図 5-11 に示したように，始状態(initial state)と終状態(final state)の違いに着目しています。

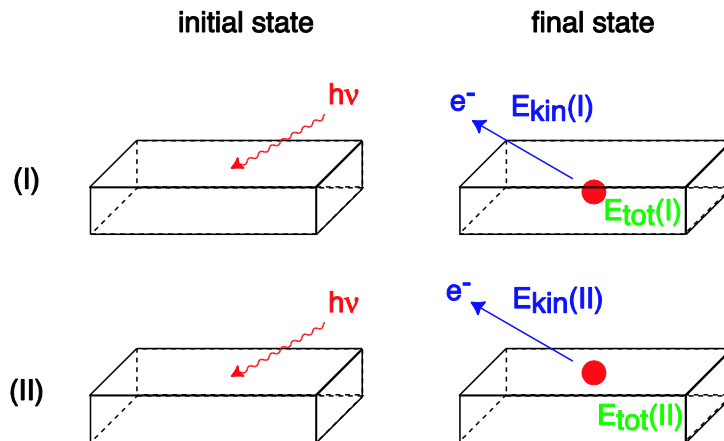


図 5-11 光電子放出過程の始状態と終状態：(I)はバルク位置の原子による光電子放出，(II)は表面付近の原子による光電子放出を表している。始状態(initial state)では，入射光($h\nu$)と結晶が存在するのみなので(I)と(II)は同じ状態である。終状態(final state)では，放出された光電子の運動エネルギー E_{kin} と内殻正孔(赤丸)1 個をもった結晶の全エネルギー E_{tot} は(I)と(II)で異なるが，和 $E_{kin} + E_{tot}$ は等しい。

図 5-11 では，光電子が放出される原子の位置が異なる 2 つの場合 を示しています。(I)は原子がバルク位置にある場合，(II)は原子が表面付近にある場合です。始状態では，考えている系には入射光($h\nu$)と結晶が存在するのみなので(I)と(II)は同じ状態です。一方，終状態では，系には放出された光電子(運動エネルギー E_{kin})と 内殻正孔(赤丸)が残された結晶(全エネルギー E_{tot})が存在します。内殻正孔とは，光電子が放出される前に占有していた内殻準位に残された電子のぬけがらです。(I)と(II)では E_{kin} と E_{tot} はそれぞれ異なるが，始状態が同一という

ことから、終状態の系の全エネルギーも同じ値にならないと次重要な関係式が導かれます。

$$E_{kin}(I) + E_{tot}(I) = E_{kin}(II) + E_{tot}(II) \quad (2)$$

内殻準位シフト ΔE_{kin} を次式で定義します。

$$\Delta E_{kin} \equiv E_{kin}(II) - E_{kin}(I) \quad (3)$$

ゆえに、式(2)を用いば

$$\Delta E_{kin} \equiv E_{tot}(I) - E_{tot}(II) = -\Delta E_{tot} \quad (4)$$

が得られます。式(2)の右边は擬ポテンシャル法でも計算することができます。Scheffler ら[2]はこの考えに基づき Si(100)表面の表面内殻準位シフトを擬ポテンシャル法で計算し、Landemark らの実験結果[1]を理論的に説明しています。内殻準位 E_c と結晶の全エネルギー E_{tot} との関係は、式(1)により

$$\Delta E_{kin} = \Delta E_c \quad (5)$$

が成り立つので、式(4)と比較して

$$\Delta E_c = -\Delta E_{tot} \quad (6)$$

が導かれます。

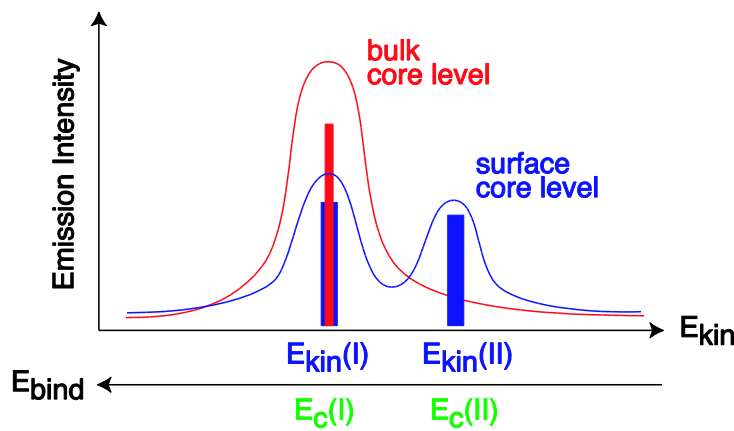


図 5-12 内殻光電子放出のスペクトル

実験的に得られる情報を模式的に表すと図 5-12 に示したようなスペクトルです。ここでは、電子間クーロン相互作用によるオージェ過程は考えません。横軸は光電子の運動エネルギー E_{kin} 、縦軸は光電子数(強度)です。また、横軸は内殻準位の結合エネルギー E_{bind} とみることもできて、この場合は左側ほど結合が強くなっています。結晶バルクからの光電子スペクトルは赤色で示したように $E_{kin}(I)$ に唯一のピークをもちます。これに対して、結晶表面付近からの光電子スペクトルは青色で示したようにいくつかのピーク $E_{kin}(II)$ に分かれます。各ピークの強度はそのピークに関係した原子数の比から定まります。電気二重層の効果のためピーク位置は通常右側にずれます。光電子は結晶内部の非弾性散乱でエネルギーを失うので、バルクのピークは結晶表面からせいぜい数 nm 程度の深さの原子によるものであることを注意してください。計算できるものは、バルク位置からピーク位置のずれと各ピークの相対的な強度です。

(2) 内殻正孔を含む原子の擬ポテンシャル

第一原理擬ポテンシャルバンド計算法では、内殻電子状態を凍結させ、価電子状態を自己無撞着に計算します。そのためバンド計算の段階では内殻に正孔を生じさせることができません。そのかわり内殻正孔を含む擬ポテンシャルを作成することは可能です。Scheffler ら[21]は、光電子放出時に生成された内殻正孔はまわりの電子によりすみやかに遮蔽(screening)されるものと仮定しました。しかし、格子を変形させるだけの時間はないとしています。また、電子はドーピングされた不純物から無尽蔵に補給される、すなわち Fermi 面は不純物準位にピンニングされる、と考えて電氣的に中性が保たれているものとしています。これらの仮定により計算されたスペクトルは実験結果とよく一致しています。

以上の考え方に従えば、内殻正孔を含む原子の擬ポテンシャルを次のように作成します。例えば $2p$ 準位に内殻

正孔を含むシリコン原子の場合、

1. 14 個の電子からなる $(1s)^2(2s)^2(2p)^6(3s)^2(3p)^2$ の電子配置の 中性シリコン原子において、 $2p$ 準位の電子 1 個を $3p$ 準位に移動して $(1s)^2(2s)^2(2p)^5(3s)^2(3p)^3$ とした励起状態の中性シリコン原子を考える
2. その原子の全電子計算を自己無撞着におこなう
3. 価電子として $(3s)^2(3p)^3$ の 5 個の電子をはがしてイオン化する
4. このようにして、内殻正孔を有し 5 価にイオン化したシリコン原子の擬ポテンシャルができる

とします。これは図 5-13 の 2 つの終状態(final state)のうち 右側の screened hole(遮蔽正孔)に対応します。

スカラー相対論での擬ポテンシャル計算では、内殻準位のスピン軌道分裂は考慮せず 縮重度に対して重み平均をとったものを内殻準位とします。内殻準位のシフト量の計算では、この仮定により結果は変わりません。

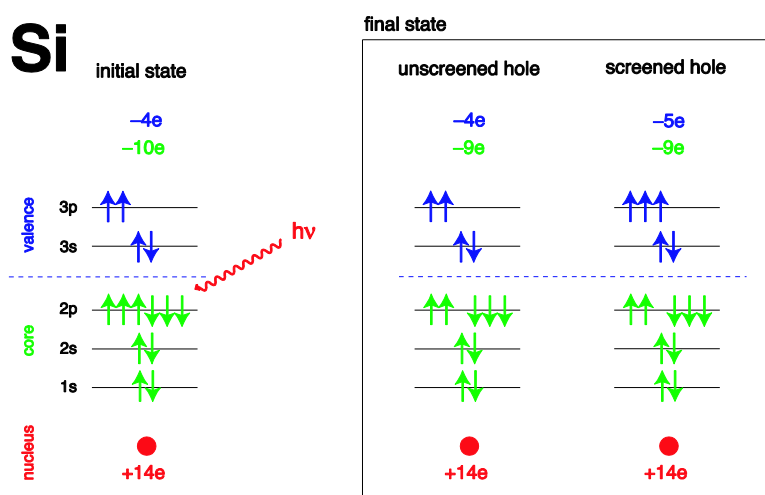


図 5-13 Si 原子の電子配置：左図は基底状態の電子配置，右図は内殻正孔が生じた場合の終状態の電子配置である。

5.1.6.2 計算の実行方法

内殻正孔を含む原子の擬ポテンシャルを用いて内殻準位シフトを バンド計算します。

バンド計算は次のように行います。

1. シリコン表面の座標を作成し、通常のシリコン擬ポテンシャルを用いて 各原子に力が働かなくなるまで十分に格子緩和させる
2. その座標を用いて、原子 1 個を内殻正孔を含む擬ポテンシャルに置き換えて 全エネルギーを計算する
3. すべての原子に対して順番に置き換えて全エネルギーを計算する
4. バルク位置の原子を決めて、これを基準にして他の配置の全エネルギーの差をとり、運動エネルギーのシフト量を計算する
5. 強度を含めたスペクトルとして表示する

5.1.6.3 計算例：Si(100)表面

(1) Si(100)表面のモデル

まずはじめに、Si(100) $p(2 \times 2)$ 再構成表面のモデルを作成します。Si(100)表面はダングリングボンド数を減らすためダイマー構造をとり、それらのダイマーは列方向に交互に傾くこと(バックリング)で安定化することが知られています。この時、バックリングに連動してダイマーの低い方から高い方に電子が移動します。実験的

には Si(100) $c(4 \times 2)$ 構造が安定となるが、計算ではバックリングの性質が 似通った $p(2 \times 2)$ 構造を扱います。この方が計算量を減らせるからです。この仮定は Scheffler ら[2]と同様です。

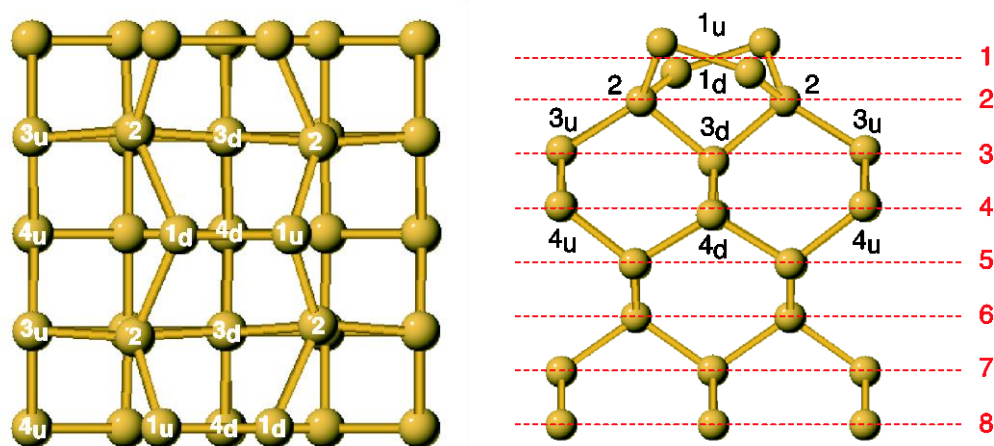


図 5-14 Si(100)表面のモデル：表面平行方向の周期は 2×2 (左図)、深さ方向は8層(右図)のモデルを採用する。8層めと9層目の中間に反転中心を考えているので、実際の計算では全部で16層64原子のスラブモデルとして扱う。単位胞の等価でない各原子に1u, 1d, 2, 3u, 3d, などのラベルをつける。

出発点の表面モデルは安定構造となつてなければならず、十分に格子緩和します。原子に働く力の最大値を 5×10^{-4} 程度以下に抑えるのに長時間を要しました。図 5-14 は格子緩和された後の安定構造を表示しています。計算条件は(原子単位), 擬ポテンシャル: Si_ggapbe_nc_01.pp, 交換相関ポテンシャル: GGA-PBE, カットオフエネルギー: $k_c(wf) = 3.5$, $k_c(chg) = 7.0$, ユニットセル: $a_1 = 14.6816015290$, $a_2 = 14.6816015290$, $a_3 = 60.0000000000$, k 点: $4 \times 4 \times 1$, です。その結果, 緩和された座標と力は以下のようになります。

	x	y	z	fx	fy	fz
1	11.654665468	7.340800747	19.731672033	-0.000108	0.000000	0.000136
2	10.943522273	0.000000002	18.308554343	-0.000156	0.000000	-0.000260
3	7.408260709	7.340800738	18.308043049	0.000156	0.000000	-0.000288
4	6.696493833	-0.000000008	19.730653256	0.000095	0.000000	0.000136
5	12.644826676	10.798805752	16.929970763	-0.000186	-0.000125	-0.000051
6	12.644826664	3.882795799	16.929970692	-0.000187	0.000125	-0.000052
7	5.707073513	11.223532172	16.928492210	0.000184	0.000135	-0.000066
8	5.707073576	3.458069376	16.928492250	0.000184	-0.000135	-0.000066
9	9.176755229	11.011875138	14.003484695	0.000002	-0.000042	-0.000291
10	9.176755074	3.669726457	14.003484706	0.000002	0.000042	-0.000291
11	1.834472402	11.011012299	14.542706201	-0.000005	-0.000002	0.000195
12	1.834472400	3.670589245	14.542706195	-0.000005	0.000002	0.000196
13	9.235196168	7.340800802	11.460786586	0.000018	0.000000	-0.000178
14	9.118076207	0.000000001	11.459447663	-0.000002	0.000000	-0.000198
15	1.882358440	7.340800774	11.863174990	0.000085	0.000000	0.000607
16	1.785205792	-0.000000009	11.863763387	-0.000075	0.000000	0.000566
17	12.980433730	7.340800764	9.066215802	0.000364	0.000000	-0.000060
18	12.895937822	0.000000005	9.076888750	0.000103	0.000000	-0.000067
19	5.455961640	7.340800779	9.081857152	-0.000145	0.000000	-0.000096
20	5.370066620	-0.000000012	9.069872950	-0.000307	0.000000	0.000025
21	12.895952884	11.013570694	6.478028955	0.000077	-0.000071	0.000099
22	12.895952919	3.668030852	6.478029019	0.000078	0.000071	0.000099
23	5.455699091	11.008682705	6.484530647	-0.000072	0.000047	0.000112
24	5.455699061	3.672918843	6.484530696	-0.000072	-0.000047	0.000112
25	9.172291984	11.011379982	3.920759253	-0.000013	-0.000002	0.000122
26	9.172291976	3.670221565	3.920759374	-0.000013	0.000002	0.000122
27	1.838705787	11.010982242	3.857185244	0.000014	0.000000	-0.000054
28	1.838705789	3.670619311	3.857185195	0.000014	0.000000	-0.000053
29	9.176000956	7.340800765	1.297682500	-0.000017	0.000000	0.002104
30	9.176000956	0.0	1.297682500	0.000472	0.000000	0.002118

31	1.835200191	7.340800765	1.297682500	-0.000774	0.000000	-0.002772
32	1.835200191	0.0	1.297682500	0.000325	0.000000	-0.002761

ここでは、反転対称を考えて半分の 32 原子の結果を示しています。第 8 層の 4 個の原子(29 番～32 番)は結晶がつぶれないように固定したので力が発生しています。

(2) 表面内殻準位シフトの計算

次に、表面内殻準位シフト(Surface Core Level Shift, SCLS)の計算を行います。格子緩和された表面モデルの原子位置を固定したまま、シリコン原子を順に 内殻正孔を含む擬ポテンシャルに置き換えて全エネルギーを計算します。

この結果を図 5-15 に示します。縦軸は光電子の運動エネルギー(単位は eV)のバルクからのずれです。この場合、バルクを 3_u 位置の原子にとっています。光電子は結晶内部で非弾性散乱されるので、あまり深い位置の原子からの光電子は実験的に観測されません。観測されるのは表面から数 nm といわれています。このため、バルクを 3_u 位置としたことは正当であると考えられます。その他、 3_u と 3_d の間にとる方法も考えられるが、図 5-14 を参考にすれば 3_u はダイマー列の外側、 3_d はダイマー列の内側にあるので 3_d からの放出強度は実験的には抑えられることが考えられるので、ここでは 3_u を バルクとしました。

図 5-15 によれば screened と unscreened の決定的な違いは 第 1 層めの down のピーク位置です。両者のちがいは電子数が unscreened の方が screened より少ないことです。そのため unscreened では down 位置のシリコン原子が電子により遮蔽されません。screened では遮蔽されるので、このちがいのために down のピーク位置が異なっただと考えられます。

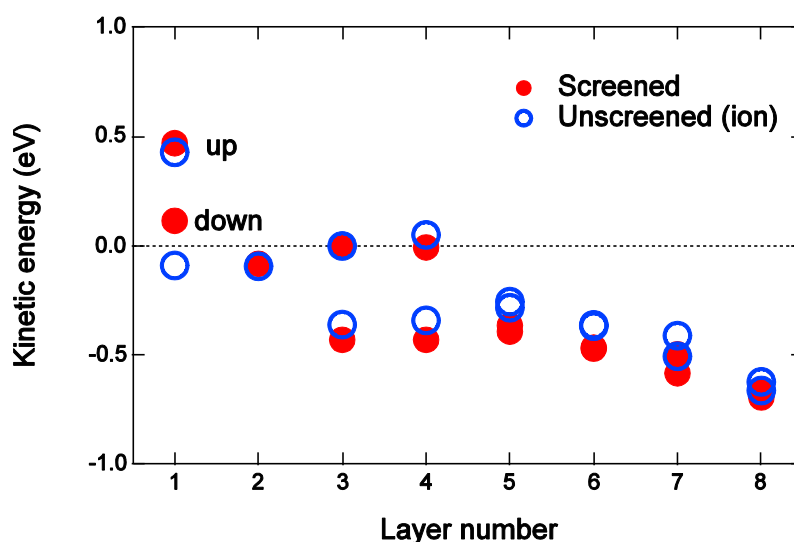


図 5-15 SCLS の運動エネルギー：それぞれ青丸は unscreened, 赤丸は screened の内殻正孔擬ポテンシャルを用いた結果である。赤丸は Scheffler と同じ方法である。縦軸は光電子の運動エネルギー(単位は eV)のバルクからのずれである。この場合、バルクを 3_u 位置の原子にとっている。

Scheffler らの論文[2]にある結果と比較するために、図 5-16 に表面から 3 層までの原子による SCLS を表示します。縦軸は強度であり、各強度は原子数の比をとっています。上から順に、unscreened, screened の各計算値、実験値を示しています。実験値は Landemark ら[1]によるものです。

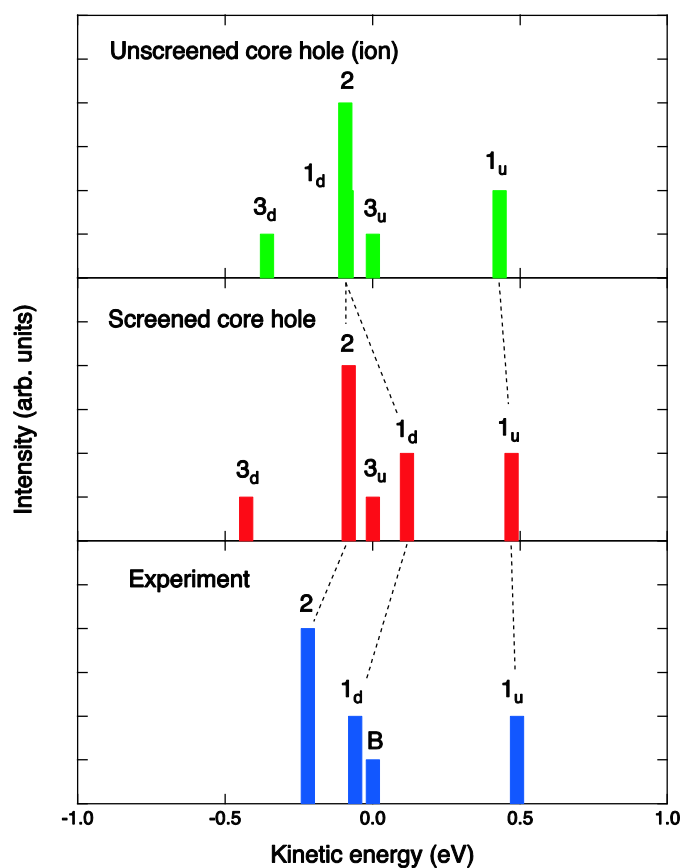


図 5-16 SCLSの強度:縦軸はSCLSの強度であり,各強度は原子数の比をとっている。上から順に, **unscreened**, **screened** の各計算値, 実験値を表示した。実験値は Landemark ら[20]によるものである。実験に合うのは **screened** である。XPS 用の内殻正孔擬ポテンシャルは **screened** で作成しなければならない。

図 5-16 の計算結果は実験結果[1]をよく再現し, また, Scheffler らの論文の結果[2]に一致しています。実験に合うのは **screened** であることがわかります。このため, XPS 用の内殻正孔擬ポテンシャルは **screened** で作成しなければなりません。

5.1.6.4 参考文献

- [1] “Core-level spectroscopy of the clean Si(001) surface: Charge transfer within asymmetric dimers of the 2×1 and $c(4 \times 2)$ reconstructions”, E. Landemark, C.J. Karlsson, Y.-C. Chao, and R.I.G. Uhrberg, Phys. Rev. Lett. **69**, 1588 (1992).
- [2] “Evidence for site-sensitive screening of core holes at the Si and Ge (001) surface”, E. Pehlke and M. Scheffler, Phys. Rev. Lett. **71**, 2338 (1993).

5.1.7 仕事関数

5.1.7.1 機能の概要

PHASE を利用して、仕事関数を評価することが可能です。ここでは、仕事関数を計算する方法を説明します。第一原理計算の枠組み内における仕事関数とは、真空準位とフェルミエネルギーとの差です。真空準位は、表面の SCF 計算を実施し、表面から十分に離れた箇所での局所ポテンシャルを利用して算出することができます。

5.1.7.2 入力パラメータ

仕事関数を計算するためには、表面のモデルを準備する必要があります。対象としたい系の、対象としたい面方位をもつ表面モデルを用意します。さらに、入力データの `postprocessing` ブロックに `workfunc` ブロックを作成し、設定を行います。

```
postprocessing{
  workfunc{
    sw_workfunc = on
    sw_add_xc_to_vloc = off
  }
}
```

各変数は以下の意味をもちます。

<code>sw_workfunc</code>	仕事関数の計算に必要なデータを出力するためのスイッチです。出力させたい場合に <code>on</code> とします。
<code>sw_add_xc_to_vloc</code>	局所ポテンシャルを出力する際に、交換相関相互作用を含めるかどうかを指定します。表面から十分離れた場所においては交換相関相互作用は 0 になると考えられるので、局所ポテンシャルに含めなくても正しい仕事関数が得られることが期待できます。デフォルト値は <code>on</code> ですが、 <code>off</code> にしておくことによってより少ない真空層で収束した仕事関数を得ることが可能です。

このような設定を行ったら、通常通り PHASE を実行します。計算が収束した後に、必要な局所ポテンシャルデータなどが出力されます。すでに収束した計算に対する継続計算として実行することも可能です。

5.1.7.3 計算の実行方法

計算が終了した段階では、局所ポテンシャルのデータが逆空間のデータとして保存されます。仕事関数を得るためには、逆空間のデータを実空間へ逆フーリエ変換し、表面内で平均を計算しその結果を出力する必要があります。このような処理を行うプログラムが `workfunc` です。このプログラムのソースコードは `src_workfunc` ディレクトリーにあります。コンパイルするためには、Fortran90 コンパイラーが必要です。`workfunc` をコンパイルするには、たとえば以下のようなコマンドを実行します。

```
% cd src_workfunc
% export F90=ifort
% make
```

環境変数 F90 に Fortran90 コンパイラーを指定します。環境変数 F90 のデフォルト値は `gfortran` です。

以下のように利用します。

```
% workfunc -z ZAXIS
```

ZAXIS に、表面に垂直とみなす軸" を指定します。*a* 軸の場合 1, *b* 軸の場合 2, *c* 軸の場合は 3 を指定します。指定しない場合のデフォルト値は 3 です。

5.1.7.4 計算結果の出力

`workfunc` の処理が終了すると、`nfvlcr.cube` と `nfvlcr_av.data` の 2 種類のファイルが生成されます。`nfvlcr.cube` ファイルは、実空間の局所ポテンシャルデータを持つ **Gaussian Cube** 形式のデータファイルです。`nfvlcr_av.data` には表面に垂直な距離と面内で平均した局所ポテンシャルのデータが記録されています。以下のようなデータ形式となっています。

```
# Fermi energy (eV) -0.37838
# distance along the z-axis(Angstrom) averaged local potential (eV)
0.104167 -0.218799E+01
0.208333 -0.250195E+01
0.312500 -0.331223E+01
0.416667 -0.427665E+01
0.520833 -0.495695E+01
0.625000 -0.496651E+01
0.729167 -0.425552E+01
.....
.....
.....
```

ファイルの 1 行目にフェルミエネルギーが **eV** 単位で記録されています。3 行目以降が実際のデータです。1 列目に \AA 単位で表面に垂直な距離が、2 列目に対応する局所ポテンシャルの面内平均 **eV** 単位で記録されます。局所ポテンシャルは、表面からある程度離れた地点においてはほぼ一定値となります。この時の値とフェルミエネルギーとの差が仕事関数に相当します。

`nfvlcr_av.data` ファイルから局所ポテンシャルがフラットになる領域を推定し、フェルミエネルギーとの差を計算することによって仕事関数をもとめる Perl スクリプトが `workfunc.pl` です。以下のように利用します。

```
% workfunc.pl nfvlcr_av.data OPTIONS
```

実行すると、計算された仕事関数の値が標準出力に出力されます。また、`workfunc.eps` という、局所ポテンシャルと表面に垂直な距離の関係をグラフ化した **EPS** ファイルも作成されます。

5.1.7.5 計算例：アルミニウムの仕事関数

アルミニウムの仕事関数の計算例を紹介します。サンプルデータは、`samples/workfunc/Al` です。

利用する系は、Al (111) 7 層の表面モデルです。表面に垂直な軸は *c* 軸とします。*c* 軸の長さは、50 \AA

としました。アルミニウム(111) 面はほとんど再構成しないので、構造最適化は施しませんでした。原点を中心に、反転対称性が存在するようにモデルを作成しました。また、交換相関相互作用は局所ポテンシャルに含めない設定で計算を行いました。アルミニウムの表面モデルは図 5-17 に示します。

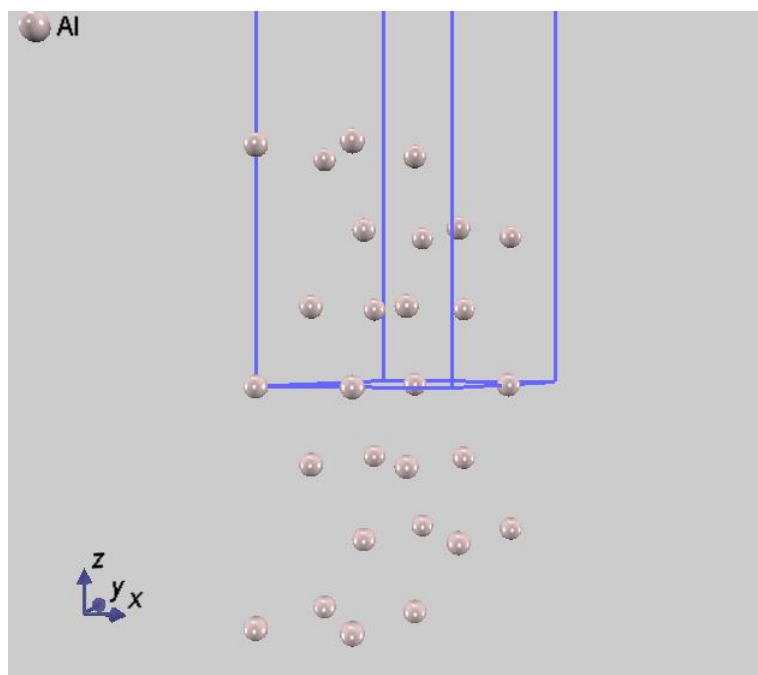


図 5-17 Al(111)面 7 層モデル

PHASE による SCF 計算が終了したのちに `workfunc` プログラムによって `nfvlcr av.data` ファイルを作成し、さらに `workfunc.pl` スクリプトを利用して得られた局所ポテンシャルと表面に垂直な距離の関係を図 5-18 に示します。仕事関数は、4.05 eV と計算されました。この値は実測値である 4.08 eV と近い結果となっています。

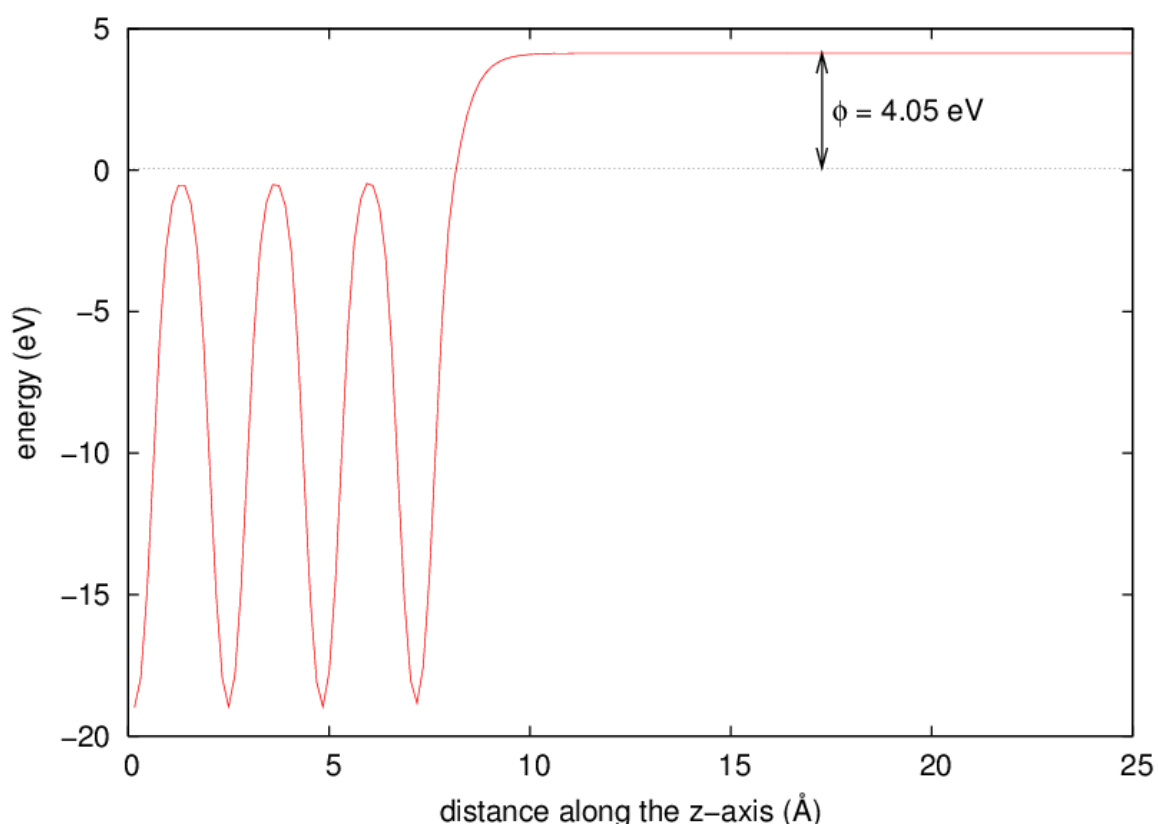


図 5-18 表面に垂直な距離と局所ポテンシャルの関係

5.1.8 陽電子寿命解析

5.1.8.1 機能の概要

電子の反粒子である陽電子は、電子と同じ質量を持ち、正の電荷を持ちます。陽電子は電子と対消滅し、 γ 線を放出します。この陽電子消滅の現象を利用して、結晶の品質や、欠陥の研究が可能です。ここで、陽電子消滅実験から有用な情報を引き出すのに、第一原理計算に基づいて陽電子寿命を予測し、実験結果と比較することが重要です。PHASE には、完全結晶における陽電子寿命を予測する機能があります。

陽電子寿命解析、以下の手順で計算を行います。

(A)	<p>はじめに通常の電子状態計算（バンド計算）を実行します。PHASE には、擬ポテンシャル・平面波法が実装されているので、この手法に基づいて計算を行います。バンド計算により、価電子の電荷密度ρ_vが得られます。全電子の電荷密度は次式で与えられます。</p> $\rho_e = \rho_v + \rho_c$ <p>ここで、ρ_cは、コア電子の電荷密度です。CIAO で作成し公開されている擬ポテンシャルのデータファイルには、自由な原子におけるコア電子の電荷密度の情報が含まれています。このデータを読み込み(1)式を評価します。</p>
(B)	<p>陽電子波動関数ψ_+は次式で与えられます（原子単位）。</p> $\left[-\frac{1}{2}\Delta - \int d\vec{r}' \frac{\rho_e(\vec{r}') - \rho_n(\vec{r}')}{ \vec{r} - \vec{r}' } + \mu_c(\rho(\vec{r})) \right] \psi_+(\vec{r}) = \varepsilon \psi_+(\vec{r})$ <p>ここで、μ_c は、電子・陽電子相関に由来するポテンシャルエネルギーであり、ρ_nは、原子核の点電荷を表します。いま、陽電子は固体中に 1 個しかないと仮定するので、最も安定な固有状態のみを求めればよいことになります。したがって、陽電子の固有状態はブリルアンゾーン中のΓ点に属します。この波動関数を平面波によって展開します。</p> $\psi_+ = \sum_{\vec{G}} C_{\vec{G}} \exp(i\vec{G} \cdot \vec{r})$ <p>ここで、逆格子周期ベクトル\vec{G}の和を有限に抑えるため、平面波の運動エネルギーの上限を設定します。</p>
(C)	<p>陽電子密度を求めます。</p> $\rho_p(\vec{r}) = \psi_+ ^2$
(D)	<p>電子及び陽電子の電荷密度を用いて、下記の式を評価し、陽電子寿命を計算します。</p> $\frac{1}{\tau} = \pi r_e^2 c \int d\vec{r} \rho_e(\vec{r}) \rho_p(\vec{r}) \Gamma(\rho_e)$ <p>ここで、r_e は、電子の古典半径、cは光速を表します。Γは、増大因子であり、電子・陽電子間の相関に由来するものです。PHASE では、上式の評価において、下記の近似を用いています。</p> $\rho_e \Gamma(\rho_e) \cong \rho_v \Gamma(\rho_v) + \rho_c \Gamma(\rho_c)$ <p>この近似が成り立つためには、価電子とコア電子の分布の重なりが小さいことが条件となります。</p>

本計算では、電子・陽電子間の相関に対して、局所密度近似を用いています。すなわち、相関ポテンシャルと増大因子は、均一な電子ガス中に陽電子が 1 個ある場合に計算された結果をもとに、電子密度の関数として与えられます。増大因子に関しては、次式が提案されています[Puska95]。

$$\Gamma = 1 + 1.23r_s + 0.9889r_s^{3/2} - 1.482r_s^2 + 0.3956r_s^{5/2} + r_s^3/6$$

ここで、 $\frac{4\pi}{3}r_s^3 = 1/\rho_e$ です。また、ギャップのある系、すなわち誘電体においては、金属よりも電子のスクリーニング効果が小さいため、次の補正を行う事を推奨します[Puska91],[Nakamoto07]。

$$\Gamma = 1 + 1.23r_s + 0.9889r_s^{3/2} - 1.482r_s^2 + 0.3956r_s^{5/2} + (1 - 1/\varepsilon_{ele})r_s^3/6$$

ここで、 ε_{ele} は電子系誘電率です。その値が実験により測定されていない場合には、UVSOR により、密度汎関数理論に基づいて評価することができます。計算手法の詳細については、文献[Nakamoto07] を参照ください。

5.1.8.2 入力パラメータ

Si 結晶中における陽電子寿命の計算を例とします。 計算例題は、sample/positron/Si です。

Si 結晶中における陽電寿命計算の入力パラメータファイルにおいて、陽電子計算に関する部分のみを説明します。

(1) Control タグで陽電子寿命計算を有効に設定

```
Control{
  positron = BULK
}
```

Control タグ中に positron=BULK と宣言すると、通常の電子状態計算(バンド計算)を行った後に、陽電子寿命計算を行います。

(2) accuracy タグで陽電子寿命計算のオプションを指定

```
accuracy{
  cutoff_pwf = 50.00 rydberg
  positron_convergence{
    num_extra_bands = 8
    delta_eigenvalue = 1.d-8 rydberg
    succession = 6
    num_max_iteration = 32000
    dtim = 0.01
    epsilon_ele = 12}
}
```

cutoff_pwf = 50.00 rydberg	これは、陽電子の波動関数を展開する際の[(3)式参照]カットオフエネルギーです。
positron_convergence {}	このタグの中で、陽電子波動関数を反復計算によって求める、すなわち(2)式を解く、際に、どのように収束解を得ようとするのか、その指定を行います。
num_extra_bands = 8	陽電子の固有状態は、基底状態 1 個のみを計算すれば充分です。しかし、反復計算で収束解を得るには、それ以外に、基底状態よりもエネルギーの高い状態の波動関数も計算する必要がある、その個数を指定します。なお、求める波動関数は全てブリルアンゾーン中の Γ 点に属します。
delta_eigenvalue = 1.d-8 rydberg	5.の説明参照
succession = 6	反復計算において、前回と今回の物理量 (7.参照) が 4.で与えられた範囲内で一致し、5.で指定された回数だけ連続して、この条件を満たせば、計算は収束したとみなします。
num_max_iteration = 32000	計算は、この数繰り返すと、収束していなくても終了します。
dtim = 0.01	繰り返し計算において、次の波動関数をどれだけ大きく変化させるかの尺度であり、dtim が大きいほど収束が早くなります。しかし、あまり大きいと収束解が得られなくなり、この値が小さいほど、安定に収束解が得られますが、小さくするほど収束が遅くなります。したがって、この値は計算する系により、ユーザーが最適な値を探すことを推奨します。
epsilon_ele = 12	epsilon_ele は、ギャップのある系に対して、LDA の電子系誘電率補正を行う際に用いる tag です。=の後には、誘電率 (Si の場合 12) を用います。もし、誘電率の補正を行わないのであれば (たとえば、金属の計算を行う場合)、8.の行は削除します。

5.1.8.3 計算結果の出力

陽電子寿命予測計算を行うと、output000 ファイルと 3 個の cube file が出力されます。

(1) ログ出力ファイル output000

このファイルの最初の部分は、Si の電子バンドの計算に関するものです。電子のバンド計算が終わり、電子の電荷密度が得られた後、陽電子の計算が行われます。

出力における“--- initial positron energy eigen values ---”からが陽電子計算に関するものです。繰り返し計算により、陽電子の波動関数が決定されます。下記の 出力は、繰り返し計算のはじめにおいて、固有値が、14.6379(eV)であることを示しています。extra bands はそれよりもエネルギーの 高い固有値 (14.9628460558-15.0292289699) を表します。繰り返し計算の 2 回目では、固有値が、0.0021898139(eV) となっています。

```
--- initial positron energy eigen values ---
=== positron eigen values ===
    14.6378982055
-- extra_bands --
    14.9628460558    14.6842242625    14.9879179620    15.2755174303
    14.8070539395    14.6061318397    14.8086346971    15.0292289699

=== positron eigen values ===
    0.0021898139
-- extra_bands --
    0.0892687578    0.1056325893    0.2037689630    0.2140559068
    0.3115605599    0.3359746459    0.3540270556    0.4738130045
```

ファイルの下の方には、次のような出力があります。

```
*****
positron lifetime(ps)   220.184723312044
core rate    3.79328791767622    %
*****
```

これは、陽電子の固有値を求める計算が収束し、陽電子の寿命が 220ps と計算されたことを示しています。core rate は、全消滅速度に対するコア電子の消滅速度の割合を示します。

(2) Cube ファイル

計算が終了すると、電子の電荷分布、陽電子の電荷分布、電子・陽電子ペアの分布が、ファイル electron.cube、positron.cube、ep_pair.cube に出力されます。これらのファイルは Gaussian cube 形式であり、可視化できます。図 5-19 に Si 結晶における計算結果を示します。価電子は主として、結合領域に存在し、陽電子は、隙間領域に存在することが分かります。陽電子の波動関数が広がり運動エネルギーが低下した方がエネルギー的に有利であることから、一般に陽電子は隙間領域に存在する傾向があります。電子・陽電子対分布を図 5-19 (c)に示します。この分布が高いところで、陽電子が大きな確率で消滅することになります。

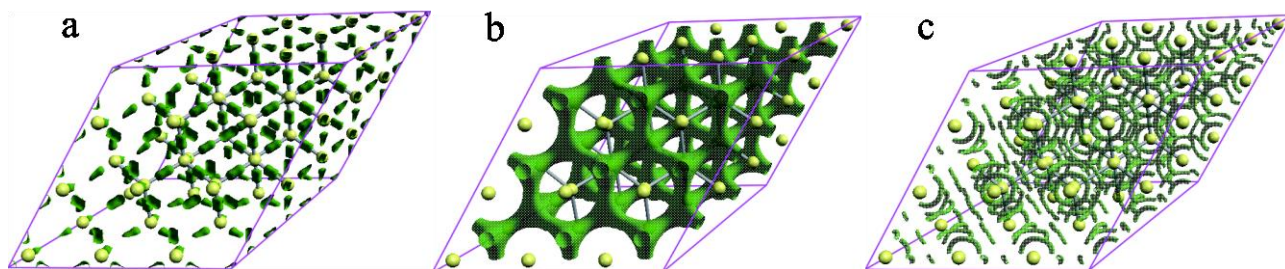


図 5-19 Si 結晶中の価電子分布(a)、陽電子分布(b)、電子・陽電子対分布(c)

5.1.8.4 使用上の注意

陽電子寿命の計算における注意点です。

- 擬ポテンシャルの選択

元素によっては、セミコア状態を持つものがあります。ここで、セミコア状態とは、コア電子の内、軌道が空間的に広がり、コア電子の分布と価電子の分布との重なりが無視できない場合のことです。この場合、セミコア電子を価電子として取り扱い、擬ポテンシャルを作成することが望まれます。公開されている擬ポテンシャルには、元素によっては、このようにして作成されたものがあり、その際はこれを利用することを推奨します。もし、そのような擬ポテンシャルが作成されていなければ、CIAO を用いて作成することもできます。

- カットオフエネルギーの選択

Si 結晶のバンド計算において、電子波動関数、電荷密度、陽電子波動関数に対するカットオフエネルギーを以下のように指定します。

```
accuracy{
  cutoff_wf = 50.00 rydberg ! cke_wf
  cutoff_cd = 200.00 rydberg ! cke_cd
  cutoff_pwf = 50.00 rydberg
```

これらの値を変化させて、計算された寿命が十分収束していることを確かめる必要があります。

- 出力

陽電子の波動関数は、電子の波動関数と同様、繰り返し計算によって求められます。各繰り返しにおいて output000 ファイルには、次のような出力があります（ここでは、繰り返しの最後における出力を表示しています）。

```
=== positron eigen values ===
-0.5674635596
-- extra_bands --
-0.0490686179 -0.0460091253 -0.0446118499 -0.0275856742
-0.0102856694 0.0069403602 0.0274419414 0.2284487012
lifetime: 220.180365487100 220.179503204077
```

ここで、計算終了近くで、陽電子の固有値 (positron eigen value) が十分収束していることを確かめます。サンプルでの出力 (output000) では、

-0.5674635596

-0.5674635638

などとなっており、十分収束していることが分かります。また、上記の出力で 220.180365487100 220.179503204077 といった数値がありますが、これは、繰り返し計算において、前回計算された、寿命と今回計算された寿命を示しており、繰り返し計算により、寿命の値が収束に近づいている事を示唆しています。

通常の電子のバンド計算が十分収束しており、かつ使用上の注意の事項を確認できれば、計算は十分考慮されたものであると考えてよいでしょう。

5.1.8.5 参考文献

- [1] M. J. Puska, A. P. Seitsonen, and R. M. Nieminen, "Electron-positron Car-Parrinello Methods: Self-consistent Treatment of Charge Densities and Ionic Relaxations", Phys. Rev. B **52** (1995) p. 10947.
- [2] M. J. Puska, "Ab-initio Calculations of Positron Annihilation Rates in Solids", J. Phys. Condens. Matter **3** (1991) p. 3455.
- [3] A. Nakamoto, M. Saito, T. Yamasaki, M. Okamoto, T. Hamada, and T. Ohno, "Two-Component Density Functional Calculations on Positron Lifetimes for Band-Gap Crystals", Jpn. J. Appl. Phys. **47** (2008) p. 2213.

5.1.9 ボルン有効電荷

5.1.9.1 概要

平面波基底を採用する第一原理計算において、原子の電荷を一意に求めることは簡単なことではありません。ボルン有効電荷はベリ一位相理論に基づいて計算されるものであり、電荷を決めるための最良の方法と考えられます。その基本的な考え方は、“原子が微小に変位した場合に発生する分極”を変位量で割ることによって電荷を求めることができる、というものです。分極も変位も3方向あり得るので、ボルン有効電荷は2階のテンソルとなります。なお、ボルン有効電荷の計算はギャップのある系でのみ計算することが可能な点にご注意ください。

5.1.9.2 ボルン有効電荷の計算の流れ

ボルン有効電荷は、次のような表式によって計算します。

$$Z_{\alpha\beta}^* = -\frac{\Omega}{q_e} \frac{\partial P_\alpha}{\partial u_\beta} = Z_{\text{ion}} \delta_{\alpha\beta} + \sum_i \frac{f}{2\pi} a_{i\alpha} \cdot \frac{\partial \phi_i(u_\beta)}{\partial u_\beta}. \quad (1)$$

ここで Z_{ion} はイオンの電荷、 $a_{i\alpha}$ は i 番目の格子ベクトルの α 成分、 ϕ_i は i 番目の逆格子ベクトルに沿ったベリ一位相、 u_β は変位の β 成分です。ベリ一位相の原子変位に対する微分は、差分近似から計算します。このことから、

3通りの変位に対して3つの逆格子ベクトルのベリ一位相を計算する必要があり、ボルン電荷を計算するためには考慮する原子数の9倍のベリ一位相計算が必要になることが分かります（実際には変位しないケースも必要なので、必要なベリ一位相計算は原子数 $\times 9 + 3$ です）。また、ベリ一位相の計算はekcalの固定電荷計算によって行うので、さらに固定電荷計算用の電荷を作成するSCF計算をベリ一位相計算に先立って行う必要があります。この一連の計算を行うために、PHASE/0に付属するberry.plというPerlスクリプトを利用します。

計算は、PHASE/0に付属するberry.plというPerlスクリプトを利用して行います。berry.plは、binディレクトリの下にあります。ボルン有効電荷の場合、berry.plは以下のような流れで計算を行います。

1. 対象とする原子すべてのSCF計算を行う。
2. 対象とする原子すべてを3方向変位させた計算を行う。
3. 1., 2.で得られた電荷密度を入力とした固定電荷計算によって、ベリ一位相計算を3つの逆格子ベクトルに対して行う。
4. 3.で得られたベリ一位相データを結合し、(1)式を利用してボルン有効電荷を計算する。

入力として必要なのは、SCF計算およびベリ一位相計算の入力の元となるテンプレート入力ファイルと、berry.plの動作を制御するコントロールファイルです。

5.1.9.3 入力データ

(1) 概要

ボルン有効電荷の計算は、複数回のPHASEによるSCF計算と、ekcalによるベリ一位相計算が必要となります。PHASE/0には、これらの計算を正しい手順で実行するberry.plというPerlスクリプトが付属しています。

berry.pl を実行するためには、SCF 計算およびベリー位相計算用のテンプレートとなる入力ファイルと、berry.pl の振る舞いを制御するコントロールファイルを作成する必要があります。

(2) SCF 計算およびベリー位相計算用テンプレート入力データの作成

ボルン有効電荷の計算を行うため、“テンプレート入力データ”を作成します。作成するテンプレート入力は、SCF 計算用とベリー位相計算用のテンプレート入力です。それぞれについて berry.pl を実行するディレクトリーの下にディレクトリーを作成し、その下に SCF 計算用の入力データと固定電荷計算用の入力データを作成します。SCF 計算は PHASE, 固定電荷計算は 2 次元版の場合は PHASE もしくは ekcal, 3 次元版の場合は PHASE プログラムを利用します。作成する入力データは通常の計算と同じですが、以下のような点に留意して入力を作成してください。

- 座標データは、構造最適化や格子最適化がなされたものを採用してください。
- 構造最適化などが行われない設定にしてください。
- 計算は、実行ディレクトリーの一階層下において行われます。すなわち、テンプレート入力が置かれるディレクトリーと同じ階層で行われます。この点に留意して file_names.data における擬ポテンシャルの指定を行ってください。
- 通常、固定電荷計算用の入力の file_names.data ファイルにおいてはファイルポインター F_CHGT を利用して SCF 計算によって得られた F_CHGT ファイル（電荷密度ファイル）を指定します。ボルン有効電荷計算においては実際に参照する SCF 計算は berry.pl が動的に決めるので、この部分を正しく指定する必要はありません（指定があっても、berry.pl によって書き換えられてしまいます）。
- 固定電荷計算で PHASE を利用する場合(3 次元版の場合必須)以下のような設定をしてください。

```
control{
    condition = fixed_charge
    fixed_charge_option{
        kparallel = one_by_one
    }
}
```

この設定を施すことによって、固定電荷計算は“k 点を一点ずつ処理する”モードで動作します。ベリー位相計算はこのモードでないと正しく計算できません。

- ベリー位相計算の **k** 点サンプリングは、berry.pl のコントロールファイルにおける mesh1, mesh2, mesh3 パラメーターによって決まるので、ekcal の入力パラメーターファイルにおける ksampling ブロックの設定は無視されます。

(3) berry.pl 用のコントロールファイルの作成

berry.pl の振る舞いは、コントロールファイルを介して指定します。ボルン有効電荷計算の場合、典型的には以下のような内容になります（#で始まる文はコメント文）。コントロールファイルのファイル名は berry.pl 実行時に指定する仕様となっているので、任意のものを採用してください。

```
#overall control
property = zeff
cpumax = 1000

#directories under which the template files reside
template_scf = scf
template_berry = berry

#parameters for the berry-phase calculation
atom_list = 1 3
displacement = 0.1

mesh1 = 6 6 15
mesh2 = 6 6 15
```

```

mesh3 = 6 6 15

#execution control
np = 4
ndir = 2
ne = 1
nk = 2
ne_b = 2
scf_command = mpiexec -np NP phase ne=NE nk=NK
berry_command = mpiexec -np NP ekcal ne=NE_B

#unit cell info, optional
a_vector = 5.01 0.0 0.0
b_vector = 0.0 5.01 0.0
c_vector = 0.0 0.0 5.01

```

この例からわかるように、パラメータ 1 つにつき 1 行を利用し、“キーワード=値”という形式でパラメータを指定します。ボルン有効電荷計算の場合に考慮する必要のあるキーワードとその説明を以下に記します。

キーワード	説明
property	どのような計算を行うかを指定します。zeff, piezo, strfrc のいずれかです。zeff を指定するとボルン有効電荷用のベリー位相計算が行われます。デフォルト値は zeff なので、ボルン電荷計算の場合未指定でも構いません。
cpumax	計算の最大時間を秒の単位で指定します。ここで指定した時間よりも経過時間が長い場合、計算はすみやかに終了します。0 以下の値を指定すると、この条件では計算は終了しません。デフォルト値は-1。
stopcheck	計算停止条件を満たしているかどうかをチェックする間隔を秒の単位で指定します。デフォルト値は 10。
length_unit	コントロールファイル中に利用される長さの単位を指定します。bohr, angstrom, nm のいずれかを指定します。デフォルト値は bohr。
template_scf	SCF 計算用のテンプレートディレクトリーのディレクトリ名を指定します。デフォルト値は template_scf。
template_berry	ベリー位相計算用のテンプレートディレクトリーのディレクトリ名を指定します。デフォルト値は template_berry。
atom_list	変位させる原子の ID を、空白区切りで指定します。
displacement	原子の変位量を指定します。property = zeff の場合に指定します。デフォルト値は 0.1 bohr。
mesh1	1 番目の逆格子ベクトルに沿ったベリー位相計算のメッシュパラメータを空白区切りで n1 n2 J のように指定します。property = zeff および piezo の場合必須の指定です。
mesh2	2 番目の逆格子ベクトルに沿ったベリー位相計算のメッシュパラメータを空白区切りで n1 n2 J のように指定します。property = zeff および piezo の場合必須の指定です。
mesh3	3 番目の逆格子ベクトルに沿ったベリー位相計算のメッシュパラメータを空白区切りで n1 n2 J のように指定します。property = zeff および piezo の場合必須の指定です。
np	MPI プロセス数を指定します。デフォルト値は 1。
ndir	ディレクトリー並列数を指定します。デフォルト値は 1。
ne	バンド並列数を指定します。デフォルト値は 1。
nk	k 点並列数を指定します。デフォルト値は 1。
ng	(三次元版のみ)G 点並列数を指定します。デフォルト値は 1。
ne_b	ベリー位相計算時におけるバンド並列数を指定します。デフォルト値は

	1。
ng_b	(三次元版のみ) ベリー位相計算時における G 並列数を指定します。デフォルト値は 1。
scf_command	SCF 計算の実行方法を指定します。たとえば, scf_command = mpirun -np NP phase ne=NE nk=NK (2D 版) scf_command = mpirun -np NP phase ne=NE nk=NK ng=NG (3D 版) などと指定します。NP, NE, NK, NG は、計算実行時に上述の np, ne, nk, ng に置き換わります。ただし、ディレクトリ並列の数によっては“あまり”が発生することがあり、その場合は ne=NE nk=NK ng=NG は省略されて計算が投入されます。デフォルト値は mpirun phase ですが、利用している環境に合わせて適切な指定をする必要があります。
berry_command	ベリー位相計算の実行方法を指定します。たとえば, berry_command = mpirun -np NP ekcal ne=NE_B (2D 版) berry_command = mpirun -np NP phase ne=NE_B ng=NG_B (3D 版) などと指定します。NE_B は上述の ne_b に、NG_B は ng_b に置き換わります。デフォルト値は mpirun ekcal です、利用している環境に合わせて適切な指定をする必要があります。
a_vector	<i>a</i> 軸の三成分を空白区切りで指定します。必須ではありませんが、指定しておくともベリー位相計算のメッシュパラメーターの参照値を算出します。
b_vector	<i>b</i> 軸の三成分を空白区切りで指定します。必須ではありませんが、指定しておくともベリー位相計算のメッシュパラメーターの参照値を算出します。
c_vector	<i>c</i> 軸の三成分を空白区切りで指定します。必須ではありませんが、指定しておくともベリー位相計算のメッシュパラメーターの参照値を算出します。

(4) 注意点

mesh1, mesh2, mesh3 パラメーターについて

ベリー位相の計算においては、対象としたい逆格子ベクトル \mathbf{b}_i に垂直な面の面積分と、 \mathbf{b}_i に沿った線積分が実行されます。この積分のメッシュは、コントロールファイルの mesh1, mesh2, mesh3 パラメーターを利用して行います。*i* 番目の逆格子ベクトルに対し、mesh*i*=n1 n2 J と空白区切りでメッシュを指定します。ここで面積分のメッシュが n1 n2, 線積分のメッシュが J です。

\mathbf{b}_i 以外の 2 つの逆格子ベクトルを \mathbf{b}_j とすると \mathbf{b}_j を \mathbf{b}_i に垂直な面に射影したベクトル, $\mathbf{b}_j - |\mathbf{b}_j| \cos \theta_{ji} \frac{\mathbf{b}_i}{|\mathbf{b}_i|}$,

が面積分のメッシュの見積りの基準となるので、その長さから決めます。線積分のパラメータは、 \mathbf{b}_i の長さをもとに決定します。コントロールファイルに a_vector, b_vector, c_vector の指定を行っておくと、この長さの計算 (bohr⁻¹ 単位) とそこから見積もられる参考のメッシュパラメーターが以下のように標準出力に出力されます (あくまで参照値であり、得られる結果の妥当性を保証するものではありません)。

```
|b_para1|, |b_para2| and |b_perp| (in bohr^-1 units)
for reciprocal vector no. 1 : 0.172224346323159, 0.107572987734313, 0.198867545420854
for reciprocal vector no. 2 : 0.172224346322494, 0.107572987734313, 0.198867545421622
for reciprocal vector no. 3 : 0.198867545420854, 0.198867545421622, 0.107572987734313

reference value for mesh parameters n1, n2 and J
```

```
for reciprocal vector no. 1 : 8, 5, 19
for reciprocal vector no. 2 : 8, 5, 19
for reciprocal vector no. 3 : 9, 9, 10
```

並列計算について

並列計算の設定について注意すべき点を挙げます。**berry.pl** による計算は、通常のパンド、**k** 点による並列(3D 版の場合さらに **G** 点並列)に加え、複数のディレクトリにまたがって並列計算を行う“ディレクトリ並列”によって行われます。したがって、ディレクトリ並列数 (パラメーター **ndir**) を 2 以上にする場合、SCF 計算の場合は **np=ne×nk** ではなく **np=ndir×ne×nk** (3D 版の場合 **np=ne×nk×ng** ではなく **np=ndir×ne×nk×ng**) となるように、ベリ一位相計算の場合は **np=ne_b** ではなく **np=ne_b×ndir** (3D 版の場合 **np=ne_b×ng_b** ではなく **np=ne_b×ng_b×ndir**) となるように並列数を調整してください。SCF 計算とベリ一位相計算とでパンド並列数が異なるのは、ベリ一位相計算は **k** 点並列に未対応のためです。また、ディレクトリ並列数を 2 以上にする場合、以下の要領で **ScaLAPACK** を無効にしてください。

```
wavefunction_solver{
  submat{
    scalapack{
      sw_scalapack = off
    }
  }
}
```

3D 版について

バージョン 2018.01 から、3D 版でもベリ一位相計算が実行できるようになりました。3D 版でベリ一位相計算を行うには、**ekcal** ではなく **phase** を使います。また、以下のような設定を施すようにしてください。

```
control{
  fixed_charge_option{
    kparallel = one_by_one
  }
}
```

5.1.9.4 berry.pl の実行

berry.pl を引数なしで実行すると、以下のようなメッセージが得られます。

```
% berry.pl
Usage : berry.pl control [OPTIONS]
```

第一引数にコントロールファイルのファイル名を指定し、さらに必要に応じてオプションを指定して制御する仕組みになっています。

以下のようなコマンドを実行すると、コントロールファイルの解釈と解析のみ行います。

```
% berry.pl control --mode=analyze
```

以下のようなコマンドを実行すると、コントロールファイルの解釈と解析のあと、計算用のディレクトリを作成します。

```
% berry.pl control --mode=gendir
```

以下のようなコマンドを実行すると、コントロールファイルの解釈と解析のあと、計算用のディレクトリを作成し、さらに計算を実行します。

```
% berry.pl control --mode=exec
```

--mode オプションのデフォルト値は gendir です。また、

```
% berry.pl --clean
```

とすると、berry.pl が作成したディレクトリーなどを一括削除することができます。

実行すると、以下のようなログを出力しながら PHASE もしくは ekcal による計算が進行します。

```
berry.pl : script to calculate the berry phase for the PHASE System
Copyright (c) 2012-2013, IIS, The University of Tokyo
script start time : Tue Mar 21 18:12:13 2017

-- parsing the control file -
...
-- generating directories --
number of SCF directories   : 7
number of berry directories : 21

-- doing SCF calculations --
running [mpiexec -n 4 $HOME/phase0/binN/phase ne=2 nk=2] under the following
directories
scf_a0
time spent in this calculation : 9 (s), total time : 9 (s)
...
...
```

まずコントロールファイルの中身が読み込まれ、その内容が出力されます。ついで、計算に必要なディレクトリーが作成されます。その後 SCF 計算が必要な回数だけ行われ、さらにベリー位相計算が必要な回数だけ行われます。実行中のコマンドと計算中のディレクトリーは、以下のような形式で出力されます。

```
running [mpiexec -n 4 $HOME/phase0/bin/phase ne=2 nk=2] under the following directories
scf_a0
```

作成される計算用ディレクトリーは、以下のようなものです。

ディレクトリー名	説明
scf_a0	参照用の、原子を変位させない系の SCF 計算が行われるディレクトリー。
scf_aaid_uuid	aid 番目の原子を、uuid の方向に変位させた系の SCF 計算が行われるディレクトリー。aid は 1 始まりであり、テンプレート入力データに記述した原子配置の順序に対応する。uuid は 1, 2, 3 のいずれかの値をとり、それぞれ x, y, z 方向に相当する。
berry_a0_ggid	参照用の、原子を変位させない系における、gid 方向の逆格子ベクトルのベリー位相計算が行われるディレクトリー。gid は 1, 2, 3 のいずれかの値をとり、それぞれ 1 番目, 2 番目, 3 番目の逆格子ベクトルに相当する。
berry_aaid_uuid_ggid	aid 番目の原子を、uuid の方向に変位させた系における、gid 方向の逆格子ベクトルのベリー位相計算が行われるディレクトリー。aid は 1 始まりであり、テンプレート入力データに記述した原子配置の順序に対応する。uuid は 1, 2, 3 のいずれかの値をとり、そ

れぞれ x, y, z 方向に相当する。 gid は 1, 2, 3 のいずれかの値をとり、それぞれ 1 番目, 2 番目, 3 番目の逆格子ベクトルに相当する。

ボルン有効電荷計算の場合、ベリ一位相計算のあとにボルン有効電荷計算用の PHASE 計算が行われます。この計算は、`zeff` ディレクトリーにおいて行われます。この計算は非並列で行われ、通常数秒から数十秒程度で終了します。

5.1.9.5 計算結果

計算が最後まで実行されると、以下のようなログが得られます。

```

--- Calculated effective charges ---
      [  2.98266  0.00512 -0.00454 ]
Zeff( 2) = [  0.00001  3.62664 -0.32666 ]
      [ -0.00010  0.27925  3.42264 ]
...
...
--- Symmetrized effective charges ---
      [  2.98266  0.00000 -0.00000 ]
Zsym( 2) = [  0.00000  3.62664 -0.32666 ]
      [  0.00000  0.27925  3.42264 ]
...
...
--- Effective charges of all atoms ---
      [  3.46565 -0.27885 -0.28289 ]
Zeff( 1) = [ -0.27885  3.14366  0.16333 ]
      [  0.24184 -0.13963  3.42264 ]
...
...
--- Averaged effective charges ---
      [  0.00151 -0.00000  0.00000 ]
Zave = [  0.00000  0.00151 -0.00000 ]
      [  0.00000  0.00000 -0.00587 ]
...
...
--- Corrected effective charges ---
      [  3.46414 -0.27885 -0.28289 ]
Zeff( 1) = [ -0.27885  3.14215  0.16333 ]
      [  0.24184 -0.13963  3.42850 ]
...
...

```

--- Calculated effective charges --- 以下に、計算された生のボルン電荷テンソルが、計算対象となった原子数分出力されます。次に、--- Symmetrized effective charges --- 以下に対称化を施したボルン有効電荷テンソルが出力されます。その次に、--- Effective charges of all atoms --- 以下に、あらわに計算しなかった原子のボルン有効電荷も含めた結果が出力されます。あらわに計算しなかった原子のボルン有効電荷テンソルは、対称性を利用して計算されます。--- Averaged effective charges --- には全原子で平均したボルン有効電荷テンソルが出力されます。全原子で平均したボルン有効電荷はすべての要素がゼロになるはずなので、この総和則を満たすように補正したボルン有効電荷テンソルが--- Corrected effective charges ---以下に出力されます。

基本的には、--- Corrected effective charges --- 以下の結果が最もよい結果となります。ただし、

対称性の与え方が適切でない場合、またはあらわに計算した原子の数が中途半端な場合に「あらわに計算しなかった原子のボルン有効電荷を、対称性を利用して計算する」ことができなくなってしまうことがあります。このような場合は、--- Calculated effective charges --- 以下の結果を利用するようにしてください。

5.1.9.6 例題

AlN のボルン有効電荷を計算する例を紹介します。

(1) 入力データ

入力データは、5.1.9.3 節で説明したように PHASE による SCF 計算用のテンプレート入力、ekcal によるベリ一位相計算によるテンプレート入力、そして berry.pl 用のコントロールファイルから成ります。

PHASE による SCF 計算用のテンプレート入力は、AlN/born/template_scf の下にあります。この入力に特殊な設定はありませんが、構造最適化がなされないよう原子の mobile 属性値はすべて off と設定されています。また、対称性を自動的に処理する機能を有効にしています。

ekcal によるベリ一位相計算のテンプレート入力は、AlN/born/template_berry の下にあります。この入力も、通常の固定電荷計算の入力と全く同じです。

berry.pl のコントロールファイルは、AlN/born/control です。その内容は以下の通り。

```
property=zeff

atom_list = 1 3
mesh1 = 6 6 15
mesh2 = 6 6 15
mesh3 = 6 6 15

scf_command = mpiexec -n NP $HOME/phase0/bin/phase ne=NE nk=NK
berry_command = mpiexec -n NP $HOME/phase0/bin/ekcal ne=NE_B
np = 1
ndir = 1
ne = 1
nk = 1
ne_b = 1
```

property = zeff とすることによって、ボルン有効電荷を計算することを指定しています。atom_list に 1 3 とすることによって対象とする原子を指定しています。1 番目の原子は Al, 3 番目の原子は N です。AlN は計 4 つの原子から成る結晶ですが、2 番目と 4 番目の原子のボルン有効電荷は対称性から求まるので計算の対象にはしていません。mesh1 mesh2, mesh3 によって逆格子ベクトルのメッシュを指定しています。この例では、すべての逆格子ベクトルに対して $n_1 = n_2 = 6$, $J = 15$ としています。scf_command, berry_command によって PHASE および ekcal の実行方法を指定しています。ここは、計算に利用する環境に合わせて適宜修正する必要があります。np, ndir, ne, nk, ne_b によって並列のさせ方を指定することができます。この例では、すべて 1, すなわち非並列で計算を行う指定となっています。

(2) 計算の実行

以下の要領で、berry.pl を実行することができます。

```
% berry.pl control --mode=exec
```

計算時間は利用する CPU やコンパイラーなどに大きく依存します。Intel Core i7-2600@3.40 GHz の CPU を搭載したマシンで実行したところ、ボルン有効電荷が得られるまで 1,760 秒かかりました。

(3) 計算結果

この計算によって得られた各原子のボルン有効電荷テンソルは、下記の通りです。

```

      [ 2.50916  0.00000 -0.00000 ]
Zeff( 1) = [ 0.00000  2.50916  0.00000 ]
      [ 0.00000  0.00000  2.64124 ]

      [ 2.50916  0.00000 -0.00000 ]
Zeff( 2) = [ 0.00000  2.50916 -0.00000 ]
      [ 0.00000  0.00000  2.64124 ]

      [ -2.50916 -0.00000  0.00000 ]
Zeff( 3) = [ -0.00000 -2.50916 -0.00000 ]
      [ 0.00000 -0.00000 -2.64124 ]

      [ -2.50916 -0.00000  0.00000 ]
Zeff( 4) = [ -0.00000 -2.50916 -0.00000 ]
      [ 0.00000 -0.00000 -2.64124 ]

```

1 番目と 2 番目の原子が Al, 3 番目と 4 番目の原子が N です。

5.1.9.7 berry.pl を使わずにボルン有効電荷を計算する方法

ボルン有効電荷の計算は, **berry.pl** を利用せずとも行うことはできます。その手続きを説明します。

(1) SCF 計算

ボルン有効電荷を計算するためには, 対象としたい原子を変位させた SCF 計算を実行する必要があります。また, 原子を変位させていない SCF 計算も実行しておく必要があります。まずは変位させない SCF 計算を通常通りに行います。ついで, 対象とする原子ごとにその位置を x, y, z 方向に微小量 (たとえば 0.1 bohr) 変位させた計算を行います。変位させる計算を行うためには, 入力パラメーターファイルの **atom_list** ブロック以下に定義できる **displacement** ブロックを利用すると便利です。

```

structure{
  ...
  ...
  atom_list{
    ...
    ...
    displacement{
      sw_displace_atom = on
      displaced_atom = 1
      ux = 0.1
      uy = 0
      uz = 0
    }
  }
}

```

sw_displace_atom = on とすると原子を変位させます。**displaced_atom** によって変位させる原子を指定します。**ux, uy, uz** によって変位の x, y, z の値を長さの指定します。

以上の入力を作成できたら, 通常通り **PHASE** を必要な回数実行します。

(2) ベリー位相計算

SCF 計算のあとに, ベリー位相計算を行います。(1) 節で行ったすべての SCF 計算をもとに, 固定電荷計算の

入力を作成します。各々の SCF 計算について、3 方向の逆格子ベクトルのベリー位相計算用入力を作成します。ベリー位相計算を実行するには、入力パラメーターファイルに以下のように最上位に **berry_phase** ブロックを作成し、設定を行います。

```
berry_phase{
  sw_berry_phase = on
  g_index = 1
  mesh{ n1 = 3, n2 = 3, J = 5 }
}
```

sw_berry_phase = on とするとベリー位相計算が有効になります。**g_index** によって対象とする逆格子ベクトルを指定します。**mesh** ブロックを作成し、**n1, n2, J** によってメッシュパラメーターを指定します。このほか、SCF 計算に対応する **displacement** ブロックを作成しておくことと、**file_names.data** において対応する SCF 計算のディレクトリーの下に電荷密度データを指すようにすることなどを忘れないようにしてください。

以上の入力を作成できたら、通常通り **ekcal** を必要な回数実行します。

(3) ボルン有効電荷の計算

最後にボルン有効電荷を計算します。以下の手続きを踏みます。

1. ボルン有効電荷計算用のディレクトリーを作成します (たとえば, **zeff**)。
2. ベリー位相データファイル **berry.data** をすべて連結し、ファイルの先頭にその数を指定します。たとえば、21 のベリー位相計算を行ったならばファイルの先頭に 21 と記述します。
3. ボルン有効電荷計算用のディレクトリーに、SCF 計算用の入力ファイルと 2. で得た **berry.data** ファイルをコピーします。
4. コピーした入力ファイルから **displacement** ブロックを削除し、以下を挿入します。

```
control{
  condition = initial
  max_iteration = 0
}
postprocessing{
  polarization{
    sw_bp_property = on
    property = effective_charge
  }
}
```

ボルン有効電荷の計算において SCF 計算を行う必要はないので、**control** ブロックにおいて **max_iteration = 0** とします。**postprocessing** 以下の設定が、ボルン有効電荷を計算するための指示です。

5. **PHASE** を非並列で実行します。結果は、**outputxxx** ファイルに、5.1.9.5 節で説明する形式で出力されます。

5.1.10 BoltzTraP を利用した解析(バージョン 2019.01 以上)

BoltzTraP[1],[2] とは、Boltzman 理論を利用し、バンド構造から輸送係数などを計算するプログラムです。必要な情報は結晶の対称性とバンドエネルギーです。**BoltzTraP** を利用すると、温度および化学ポテンシャルの関数として以下のような量を求めることができます。

- ゼーベック係数
- 伝導度 (緩和時間で割った値が出力される)
- ホール係数
- 熱伝導率 (電子からの寄与分のみ ; 緩和時間で割った値が出力される)
- 電子比熱
- 磁化率

ここでは、PHASE/0 が計算するバンド構造から BoltzTraP を利用する方法を説明します。BoltzTraP にはその使い方が大きく異なるバージョン 1. とバージョン 2. がありますが、PHASE/0 はどちらのバージョンにも対応しています。

5.1.10.1 入力ファイル

BoltzTraP が必要とするファイルを出力するには、入力パラメーターファイルの Postprocessin ブロックにおいて boltztrap ブロックを作成し、設定を施します。

```
Postprocessing{
  boltztrap{
    sw_boltztrap = on
    version = 2
    prefix = CoSb3
    header = "CoSb3 calculated by PHASE/0"
  }
}
```

boltztrap で定義できるタグは下記の通り。

タグ名	説明
sw_boltztrap	BoltzTraP で利用できるデータファイルを出力するかどうかを指定するスイッチ。on とするとデータファイルを出力する。
version	BoltzTraP のバージョンを指定する整数。1 もしくは 2 を指定する。デフォルト値は 1.
prefix	BoltzTraP 用データファイルの接頭辞を指定する文字列。デフォルト値は実行ディレクトリー名。
header	BoltzTraP 用データファイルのヘッダーを指定する文字列。デフォルト値は prefix と同じ値。

5.1.10.2 PHASE/0 の実行

PHASE/0 の実行は通常通り行います。計算機能の制限などは特にありません。構造最適化、格子最適化が収束すると BoltzTraP で必要なデータファイルが出力されます。SCF 計算でも固定電荷計算でも BoltzTraP による計算は可能ですが、BoltzTraP の解析は k 点数に対する収束性が悪い場合があり、 k 点数を変えながら収束を確かめる、という手続きを踏む必要があります。そのため、固定電荷計算を活用することを推奨します。

5.1.10.3 BoltzTraP の実行

ここでは、BoltzTraP による解析を実施する方法を簡単に説明します。詳細は BoltzTraP のマニュアル、チュートリアルをご覧ください。なお、ここでの説明は BoltzTraP のインストールは終了していることを前提としています。

(1) バージョン 1.

バージョン 1. を実行するために必要なファイルは *prefix.intrans*, *prefix.struct*, *prefix.energy* の 3 つです。この 3 つのファイルはいずれも PHASE/0 から出力されますが、*prefix.intrans* ファイルは BoltzTraP の動作を制御する“入力ファイル”なので実行したい解析に応じてユーザーが編集する必要があります。*prefix.intrans* の内容は、典型的には下記のようになっています。なお、行頭の数値は説明用なので、実際には記述しないでください。

```
1:  GENE                      # Format of DOS
2:  0 0 0 0.0                 # iskip idebug setgap shiftgap
3:  0.17267 0.0005 0.4 58     # Fermilevel (Ry), energygrid, ...
4:  CALC                      # CALC (calculate expansion coeff), NOCALC
5:  5                          # lpfac, number of latt-points per k-point
```

6:	BOLTZ	# run mode (only BOLTZ is supported)
7:	.15	# (efcut) energy range of chemical potential
8:	800. 50.	# Tmax, temperature grid
9:	-1.	# energyrange of bands
10:	TETRA	
11:	0 0 0 0 0	# For scattering model undocumented
12:	2	# number of
13:	1E20 -1E20	# fixed doping levels in cm-3

この入力の#以降はコメントとなっています。また、ファイルは固定形式なので、記述された数値や文字列を削除してしまうとエラーになります。

重要と思われる（変更する機会が比較的多い）設定値を赤文字で示しています。2 行目後半 2 つの数値は、バンドギャップのシフトを行うかどうかを指定します。3 つめの数値に 1 を指定するとバンドギャップの補正が行われます。4 つめの数値にシフト量を Rydberg 単位で指定します。7 行目は、考慮する化学ポテンシャルの幅を Rydberg 単位で指定します。8 行目は、考慮する温度を指定します。まず最高温度を指定し、次いで温度の刻み幅を指定します。0K から最高温度まで、温度の刻み幅で分割した温度について計算がなされます。12 行目、13 行目では考慮するドーピングレベルを指定します。まず 12 行目でドーピングレベルの数を指定し、さらに 13 行目でドーピング濃度を cm^{-3} 単位でドーピングレベル数分指定します。なお、ドーピングを考慮するといっても特別な計算が別途行われるわけではありません。ドーピングレベルに応じた化学ポテンシャルによってデータを並べ替え、結果を出力してくれるという計算機能です。

prefix.intrans ファイルを望みのものに変更したら、以下の要領で BoltzTraP を実行します。PATH_TO_BOLTZTRAP は BoltzTraP をインストールしたディレクトリーに読み替えてください。

%PATH_TO_BOLTZTRAP/src/x_trans BoltzTraP (スピン軌道を考慮していない場合)
%PATH_TO_BOLTZTRAP/src/x_trans BoltzTraP -so (スピン軌道を考慮している場合)

なお、BoltzTraP の計算結果は k 点数をある程度多くしないと収束しません。結果が収束しているかどうか、 k 点数を変えた計算を行い確認することが推奨されます。BoltzTraP のマニュアルによると、目安として $16 \times 10^6 / V$ (V は単位胞の体積) 程度の k 点を取ることが推奨されています。

BoltzTraP による計算が終了すると、下記のようないろいろなファイルが結果として出力されます。

ファイル名	説明
<i>prefix.outtrace</i>	ログファイル
<i>prefix.trace</i>	各種テンソルの trace が出力される
<i>prefix.condtens</i>	伝導テンソルの全要素が出力される
<i>prefix.halltens</i>	ホール係数テンソルの全要素が出力される
<i>prefix.trace_fixdoping</i> , <i>prefix.condtens_fixdoping</i> , <i>prefix.halltens_fixdoping</i>	ドーピングを考慮している場合に、 <i>prefix.xx</i> の内容を濃度に合わせて加工したデータが出力される

これらのファイルの形式については、BoltzTraP のマニュアルを参照してください。

(2) バージョン 2.

BoltzTraP バージョン 2 は Python のモジュール群であり、Python スクリプトの中で利用できるように設計されていますが、btp2 というフロントエンドとなるコマンドも用意されています。btp2 コマンドに様々な副コマンドやオプションを渡すことによって BoltzTraP バージョン 2 による計算を行うことができます。ここでは、このコマンドの使い方について説明します。

`bt2` を実行するために必要なファイルは `prefix.energy` と `prefix.structure` ファイルです。これらのファイルが置かれたディレクトリーを `prefix` とすると、`prefix` の親ディレクトリーにおいて以下の要領で“フーリエ補間”を行います。

```
% bt2 interpolate -m 5 prefix
```

`interpolate` は“フーリエ補間を行う”ことを `bt2` に伝える副コマンド、`-m 5` は k 点 1 点あたり 5 点の補間を行う、というオプションです。これによって `interpolation.bt2` というファイルが作成されます。この補間の結果が記録されたファイルを入力として様々な解析が行えるようになっています。たとえば、ある温度範囲で輸送係数の計算を行いたい場合以下のようなコマンドを利用します。

```
% bt2 integrate interpolation.bt2 300:500:1
```

`integrate` は“輸送係数の計算を行う”ことを `bt2` に伝える副コマンドです。そのあとに作成した補間の結果ファイル `interpolation.bt2` を渡し、さらに温度範囲を最低温度:最高温度:温度の刻み幅という形式で指定しています。得られるファイルなどについては、`BoltTraP` のバージョン 1. と違いはありません。

5.1.10.4 計算例

体心立方 CoSb_3 結晶のゼーバック係数の計算例を紹介します。 CoSb_3 の原子配置は図 5-20 に示すとおり。

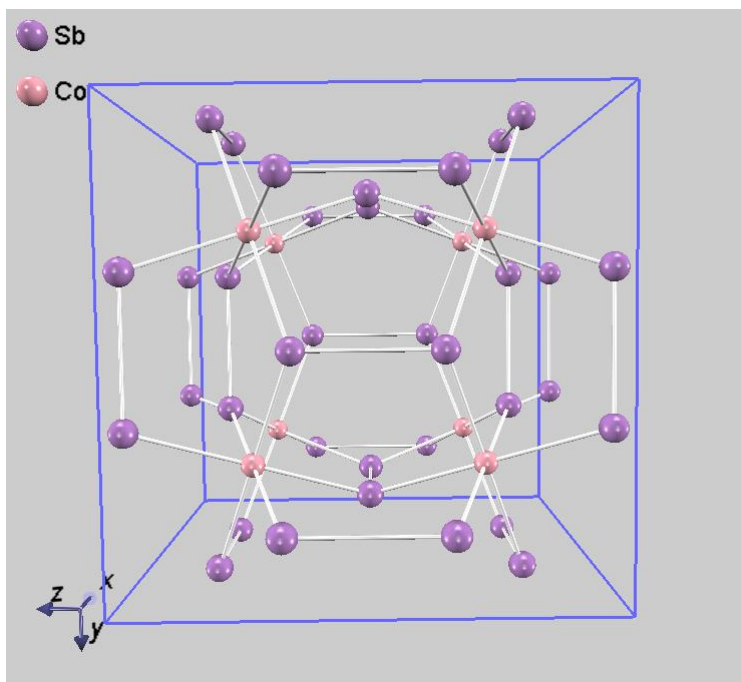


図 5-20 CoSb_3 の結晶構造 (体心立方構造)

なお、この例題の入力ファイルは `samples/boltztrap` 以下に置かれています。

まず SCF ディレクトリーにある入力ファイルを用いて通常の SCF 計算を行います。この SCF 計算においては k 点サンプリングは $4 \times 4 \times 4$ の Monkhorst-Pack メッシュを採用しています。つづいて、`fc_5x5x5`, `fc_10x10x10`, `fc_20x20x20`, `fc_30x30x30` ディレクトリーにおいて k 点メッシュを変更した固定電荷の計算を行います。固定電荷計算の入力ファイルには、以下のように `BoltTraP` が読み込める形式の固有値データなどが出力されるような設定が施されています。

```
postprocessing{  
  boltztrap{
```

```

    sw_boltztrap = on
    version = 1
    header = "CoSb3 ksamp 10x10x10 bands 96"
  }
}

```

ekcal もしくは phase による固定電荷計算を行うと、以下の 3 つのファイルが得られるはずです (fc_5x5x5 ディレクトリーの場合)

```
fc_5x5x5.energy fc_5x5x5.struct, fc_5x5x5.intrans
```

*.energy, *.struct ファイルは固有エネルギーと座標データが記録されているファイルですが、通常編集する必要はありません。*.intrans は BoltzTraP の入力ファイルなので目的によっては編集する必要があるかもしれませんが、編集せずとも BoltzTraP による基本的な計算は行うことができるようになっています。

BoltzTraP は以下の要領で実行できます (BoltzTraP の実行ファイルである x_trans にパスが通っていると仮定)。

```
$ x_trans BoltzTraP
```

これによって、ディレクトリー名を接頭辞としてもつ多くのファイルが出力されます。ゼーベック係数などの輸送係数の計算結果が出力されるのが *.trace ファイルです。たとえば、fc_5x5x5 の場合その内容は下記のようになります。

#	Ef[Ry]	T [K]	N	DOS(Ef)	S	s/t	R_H	kappa0	...
0.02274	50.0000		34.92246648	0.45246722E+03	-0.16967269E-04	0.94428193E+20	0.15114046E-08	...	
0.02274	100.0000		34.89576852	0.46772674E+03	-0.82371993E-05	0.95266319E+20	0.11470114E-08	...	
0.02274	150.0000		34.86708620	0.47412414E+03	-0.53033332E-05	0.94512268E+20	0.10363499E-08	...	
0.02274	200.0000		34.83810551	0.47509694E+03	-0.38465950E-05	0.94664477E+20	0.93685045E-09	...	
0.02274	250.0000		34.80983692	0.47363201E+03	-0.18452224E-05	0.95321416E+20	0.85951902E-09	...	
...									
...									

結果は、同じ化学ポテンシャルのデータがセットで出力されます。ゼーベック係数は 5 カラム目のデータです。300K におけるゼーベック係数を、 k 点サンプリングを変化させながら計算した結果を図 5-21 に図示します。この結果から、 $5 \times 5 \times 5$ のケースはそのほかの場合と比較し明らかに異なる傾向を示しているため、不十分な k 点サンプリングであるといえます。 $10 \times 10 \times 10$, $20 \times 20 \times 20$, $30 \times 30 \times 30$ はおおよそ一致していますが、 $10 \times 10 \times 10$ のケースではたとえば化学ポテンシャルが 0.25 から 0.3 Rydberg の領域で異なっているように見えます。 $20 \times 20 \times 20$, $30 \times 30 \times 30$ はほぼ見分けがつきません。したがって、この例では $20 \times 20 \times 20$ のサンプリングでほぼ収束した結果が得られていると考えられます。

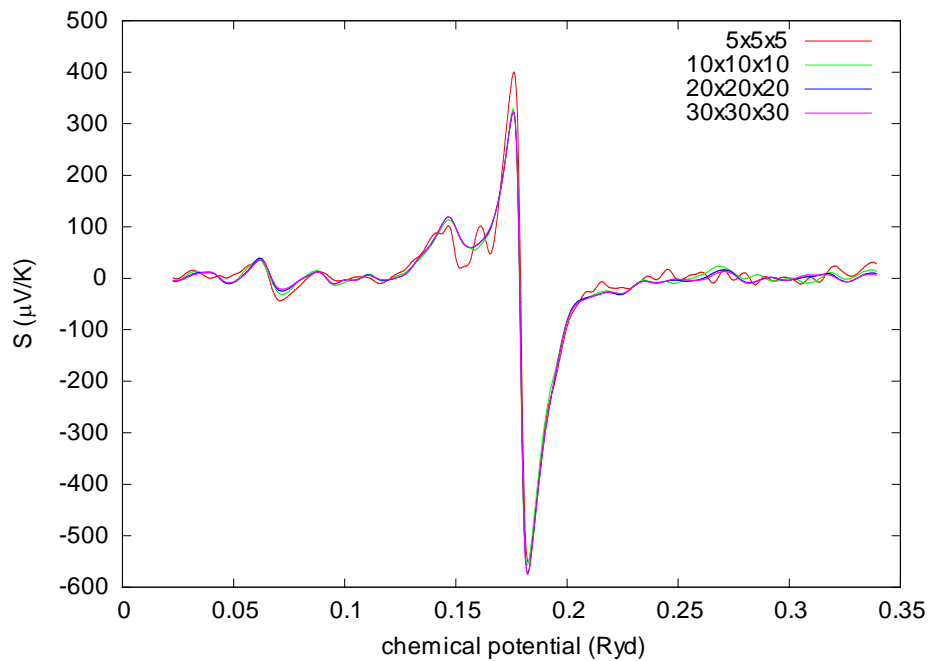


図 5-21 CoSb₃ の 300K におけるゼーベック係数

5.1.10.5 参考文献

- [1] Georg K.H.Madsen and David J.Singh, “BoltzTraP. A code for calculating band-structure dependent quantities”, Computer Physics Communications **175**, 67, (2006).
URL: https://www.imc.tuwien.ac.at/forschungsbereich_theoretische_chemie/forschungsgruppen/prof_dr_gkh_madsen_theoretical_materials_chemistry/boltztrap/
- [2] Georg K.H.Madsen, JesúsCarrete, and Matthieu J.Verstraete, “BoltzTraP2, a program for interpolating band structures and calculating semi-classical transport coefficients”, Computer Physics Communications **231**, 140, (2018).
URL: <https://www.imc.tuwien.ac.at/index.php?id=21094>

5.2 原子ダイナミクス

5.2.1 振動解析

5.2.1.1 機能の概要

PHASE には格子振動の基準モードを計算する振動解析機能があります。まず、原子を平衡位置からわずかに変位させて力計算を行います。その力から力定数行列を計算し、それから動力学行列を計算します。動力学行列の固有値問題を解くことにより、基準振動の振動数と固有ベクトルを計算します。

i 番目の原子の安定位置から変位を \mathbf{u}_i とします。変位が微小で二次以上の項が無視できるとき、格子系の運動方程式は

$$m_i \ddot{u}_{i\alpha} = - \sum_{j\beta} \Phi_{i\alpha,j\beta} u_{j\beta}$$

と書けます。 $\Phi_{i\alpha,j\beta}$ は力の定数で、原子変位に関する系のエネルギー $E(\mathbf{u}_1, \mathbf{u}_2, \dots)$ の二階微分として定義されています。

$$\Phi_{i\alpha,j\beta} = \frac{\partial^2 E}{\partial u_{i\alpha} \partial u_{j\beta}}$$

力の定数はヘルマン-フラインマン力を原子変位で微分することにより、求めることができます。

$$\Phi_{i\alpha,j\beta} = - \frac{\partial F_{i\alpha}}{\partial u_{j\beta}}$$

本プログラムでは、この微分は中央差分近似で行われます。変位パラメータを a とすると

$$\frac{\partial F_{i\alpha}}{\partial u_{j\beta}} = \frac{F_{i\alpha}|_{u_{j\beta}=a} - F_{i\alpha}|_{u_{j\beta}=-a}}{2a}$$

と書けます。力の定数には結晶の対称性による制約があり、これを満たすように力の定数を補正する必要があります。第一に、 i, j 原子が空間群の対称操作 $\{R|\mathbf{T}\}$ で i', j' 原子に移るとき、力定数テンソル Φ_{ij} は力定数テンソル $\Phi_{i'j'}$ を回転行列 R で回転させたものに等しいです。つまり、

$$R\mathbf{r}_i + \mathbf{T} = \mathbf{r}_{i'}$$

$$R\mathbf{r}_j + \mathbf{T} = \mathbf{r}_{j'}$$

ならば、

$$\Phi_{ij} = R^T \Phi_{i'j'} R$$

でなければなりません。第二に、力定数テンソル Φ_{ij} の成分 $\alpha\beta$ を原子の番号 j すべてにわたり足し合わせると、ゼロになります。つまり、

$$\sum_j \Phi_{i\alpha,j\beta} = 0$$

です。第三に、力定数行列は対称でなければなりません。つまり、

$$\Phi_{i\alpha,j\beta} = \Phi_{j\beta,i\alpha}$$

です。換算変位 $w_{i\alpha} = u_{i\alpha} \sqrt{m_i}$ と 動力学行列 $D_{i\alpha,j\beta} = \Phi_{i\alpha,j\beta} / \sqrt{m_i m_j}$ を用いて、格子系の運動方程式(10)を

$$\ddot{w}_{i\alpha} = - \sum_{j\beta} D_{i\alpha,j\beta} w_{j\beta}$$

と書き換えます。この方程式を解くために、 $w_{i\alpha} = Q \xi_{i\alpha} e^{i\omega t + \delta}$ という解を仮定します。

$$\omega^2 \xi_{i\alpha} = \sum_{j\beta} D_{i\alpha,j\beta} \xi_{j\beta}$$

これは固有値が ω^2 で、固有ベクトルが $\xi_{i\alpha}$ となる、行列 $D_{i\alpha,j\beta}$ の固有値問題です。振動解析機能ではこの固有値問題を解き、格子振動の基準モードを求めます。

5.2.1.2 入力パラメータ

振動解析を行うには、まず原子が平衡位置にある必要があります。平衡状態にないと動力学行列の固有値が負になり、振動数が純虚数のソフトモードが現れます。平衡位置の原子座標は、構造最適化機能を用いて計算します。構造最適化計算が終了したら、`nfdynm.data` の最後に記述されている最適構造での入力パラメータファイルを作成します。

振動解析の設定は、**Phonon** ブロックで指定します。

```
Phonon{
  sw_phonon = on
  sw_calc_force = on
  sw_vibrational_modes = on
  displacement = 0.05
}
```

振動解析の入力変数を以下に示します。

振動解析に関する変数の説明

変数名またはタグ名	デフォルト値	説明
<code>sw_phonon</code>	OFF	格子振動解析設定ブロックを有効にするかどうかのスイッチです。
<code>sw_calc_force</code>	OFF	振動解析のための力計算を行うかどうかのスイッチです。 ON：格子振動解析のための力計算を行います。(計算した力は <code>force.data</code> に出力されます。) OFF： <code>sw_vibrational_modes=ON</code> ならファイル "F_FORCE" から力のデータを読み込みます。
<code>displacement</code>	0.1	原子変位パラメーター。
<code>sw_vibrational_modes</code>	OFF	格子振動解析を行うかどうかのスイッチです。 ON：格子振動解析が行われ、 <code>mode.data</code> ファイルに結果が出力されます。 OFF：格子振動解析は行われません。
<code>norder</code>	1	差分次数を変更するパラメーターです。
<code>sw_polynomial_fit</code>	OFF	ON：多項式フィットで力の微分を求めます。 OFF：差分で力の微分を求めます。

● 原子座標と対称性の入力

原子座標は反転対称があってもすべて入力する必要があります。すなわち、原子座標の `weight` 属性値による省

略は利用できません。また、`sw_inversion` は OFF とする必要があります。振動モードの分類と入力座標の対称性チェックに系の空間群を使用するので、結晶構造またはその空間群を `symmetry` ブロックで正しく指定します。ただし、対称性自動判定機能を利用することも可能です。

- 元素の質量の指定

元素の質量は `element_list` ブロックの変数 `mass` で指定する。原子単位(a.u.)ではなく、原子質量単位(amu)で入力するには、`#units atomic_mass` を `#tag` 行の上に挿入する。

- 原子変位の選択

原子変位は `Phonon` ブロックの `displacement` で設定します。通常、原子変位は 0.1 a.u.以下にとると良いです。振動数の原子変位依存性を調べて、希望する振動数の収束が得られる原子変位に設定します。`norder` を 2 に設定することで、差分の次数を 3 から 5 に換えることができます。`displacement` で設定した値を `u` とすれば、原子変位は `-u, -u/2, u/2, u` になります。`sw_polynomial_fit` を ON にして多項式フィットにすれば、`norder` を 2 より大きく設定できます。そのときの原子変位は `-u/norder, -u/(norder-1), ..., u/(norder-1), u/norder` です。`norder` を大きくすると微分精度はよくなりますが、力計算の回数が増えるので、計算時間は `norder` が 1 の場合の $2^{*norder}$ 倍になるので注意してください。

5.2.1.3 計算結果の出力

振動解析結果は、振動解析結果ファイル `mode.data`、力のデータ `force.data` に出力されます。

`mode.data` には振動解析の結果が記述されます。まず最初に基本並進ベクトル $\mathbf{a}_i = (a_{ix}, a_{iy}, a_{iz})$ が次の形式で記述されます。

```
--- primitive lattice vectors ---
a_1x a_1y a_1z
a_2x a_2y a_2z
a_3x a_3y a_3z
```

次に原子の数 `natm` と各原子の座標 (x_i, y_i, z_i) と質量 m_i とラベル `name(i)` が次の形式記述されます。

```
--- Equilibrium position and mass of each atom---
Natom = natm
do i=1,natm
  i  x(i)  y(i)  z(i)  m(i)  name(i)
end do
```

次に振動解析の結果が次の形式で記述されます。

```
--- Vibrational modes ---
Nmode= nmode  Natom= natm
do m = 1,nmode
  n=  m  representation(m)  acvtive(m)
  hbarW= omega_ha(m)  Ha = omega_ev(m)  eV;  nu= omega_nu(m)  cm^-1
  do i=1,natm
    i  vec(m,i,1)  vec(m,i,2)  vec(m,i,3)
  end do
end do
```

`representation` は既約表現の配列です。`active(m)` はラマン活性なモードあれば `R` になり、赤外活性なモードであれば `IR` となります。両活性であれば、`IR&R` となります。サイレントモードの場合には何も表示されません。`vec` は固有ベクトルの配列で、`omega_ha` は Hartree 単位での振動数で、`omega_ev` は電子ボルト単位での振動数で、`omega_nu` は波数です。

力データファイル **force.data** には力の定数を計算するための力のデータが記述されます。その力データは次の形式で出力されます。

```
num_force_data, norder, sw_ploynomial_fit
do i = 1, num_force_data
  displaced_atom, displacement(1:3)
  do ia = 1, natm
    i, force_data(ia,1:3,i)
  end do
end do
```

num_force_data は力を計算する原子配置の数で, **displaced_atom** は変位した原子の番号で, 配列 **displacement** が原子の変位ベクトル(u_x, u_y, u_z)です。 **norder** は入力で指定した **norder** の値が記述されます。**sw_ploynomial_fit** は入力の **sw_ploynomial_fit** が ON のときに, ON を表す 1 が記述されます。OFF の場合には, 0 が記述されます。

sw_calc_force を OFF に設定することで, 出力された力を読み込み, 振動解析をやり直すことができます。元素の質量を変更することは問題ありませんが, 力計算に関連する変数は変更してはなりません。

5.2.1.4 計算例：水分子の振動解析

(1) 構造最適化

振動解析を行うには原子が平衡状態になければなりませんので, 振動解析を行うときと同じ条件で構造最適化を行います。平衡状態にないと動力学行列の固有値が負になり, 振動数が純虚数のソフトモードが現れます。水分子の構造最適化の入力例を以下に示します。

```
control{
  condition = initial
  cpumax = 1 day ! maximum cpu time
  max_iteration = 6000
}

accuracy{
  cutoff_wf = 25.00 rydberg
  cutoff_cd = 225.00 rydberg
  num_bands = 8
  xctype = ggapbe
  initial_wavefunctions = matrix_diagon
  matrix_diagon {
    cutoff_wf = 5.0 rydberg
  }
  ksampling{
    method = gamma
  }
  scf_convergence{
    delta_total_energy = 1.e-10
    succession = 3
    num_max_iteration = 300
  }
  force_convergence{
    max_force = 1.e-4
  }
  initial_charge_density = Gauss
}

structure{
```

```

unit_cell_type = primitive
unit_cell{
  a_vector = 15.0          0.0          0.0
  b_vector =  0.0          15.0          0.0
  c_vector =  0.0          0.0          15.0
}
symmetry{
  tspace{
    lattice_system = primitive
    generators{
      #tag rotation tx ty tz
      C2z      0 0 0
      IC2x     0 0 0
    }
  }
  sw_inversion = off
}

magnetic_state = para

atom_list{
  coordinate_system = cartesian
  atoms{
    !#default mobile=on
    !#tag rx          ry          rz          element
        -1.45        0.000      1.123        H
         1.45        0.000      1.123        H
         0.0         0.0        0.0         O
  }
}
element_list{ #units atomic_mass
  #tag element  atomicnumber  zeta  dev
        H           1         1.00  0.5
        O           8         0.17  1.0  }
}

wf_solver{
  solvers {
    !#tag sol      till_n  dts  dte  itr   var      prec  cmix  submat
        msd        5      0.1  0.1   1     tanh  on   1    on
        lm+msd     10     0.1  0.4   50    tanh  on   1    on
        rmm2p      -1     0.4  0.4   1     tanh  on   2    on
  }
  rmm {
    edelta_change_to_rmm = 1.d-6
  }
  lineminimization {
    dt_lower_critical = 0.1
    dt_upper_critical = 3.0
  }
}

charge_mixing{
  mixing_methods {
    !#tag id method  rmxs  rmxe  itr  var      prec  istr  nbxmix  update
        1  broyden2  0.3   0.3   1   linear  on   5     10     RENEW
        2  simple   0.2   0.5  100  linear  on   *     *      *
  }
}

```

```
structure_evolution{
  method = cg
}
```

file_names.data には element_list と同じ順番でポテンシャルファイル H_ggapbe_nc_01.pp と O_ggapbe_us_02.pp を指定します。この入力を使用して得た水分子の構造を図 5-22 に示します。

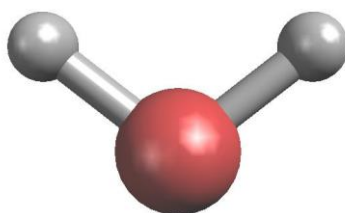


図 5-22 水分子の構造

(2) 振動解析

構造最適化後に振動解析を行うには、入力の原子座標を最適化した座標に換えて、Phonon ブロックを加え、振動解析の設定をします。最適化原子座標は構造最適化計算の出力ファイル nfdynm.data に記述されている最後のステップの原子座標です。

```
atom_list{
  coordinate_system = cartesian
  atoms{
    !#tag  rx      ry      rz      element
      -1.446816228  0.000  1.123327795  H
       1.446816228  0.000  1.123327795  H
        0.0         0.0    0.0         O
  }
}
```

振動解析の設定はたとえば以下のようにします。原子変位は0.05 とします。

```
Phonon{
  sw_phonon = on
  sw_calc_force = on
  sw_vibrational_modes = on
  displacement = 0.05
}
```

PHASE を実行します。

```
% mpirun ../.../bin/phase
```

PHASE を実行すると、振動解析結果のファイル mode.data が出力されます。

振動数レベル図はツール freq.pl を使用して作成します。分子の場合には以下のように-mol というオプションを

付けて **freq.pl** を実行します。

```
% freq.pl -mol mode.data
```

この例題の水分子の基準モードの振動数を図 5-23 に示します。

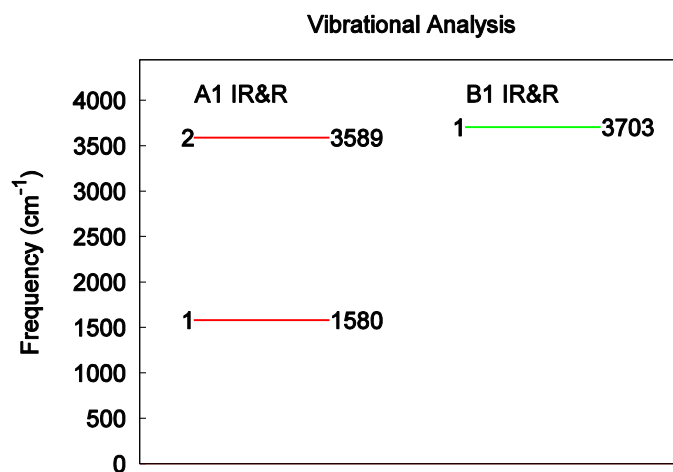


図 5-23 水分子の振動モードの振動数

基準振動の固有ベクトルの図を作成するための拡張 **trajectory** 形式のファイルは、ツール **animate.pl** で作成します。原点の移動を指定したファイル **control.inp** を用意します。**control.inp** は以下のように記述します。

```
origin 7.5 7.5 7.5
```

ツール **animate.pl** を以下のように実行します。

```
% animate.pl mode.data control.inp
```

基準振動の固有ベクトルの拡張 **trajectory** 形式のファイル **mode_*.tr2** が生成されます。

この例題の水分子の基準モードの固有ベクトルを図 5-24 に示します。生成された振動モードの拡張 **trajectory** 形式のファイル **mode_7.tr2, mode_8.tr2, mode_9.tr2** を可視化したものです。

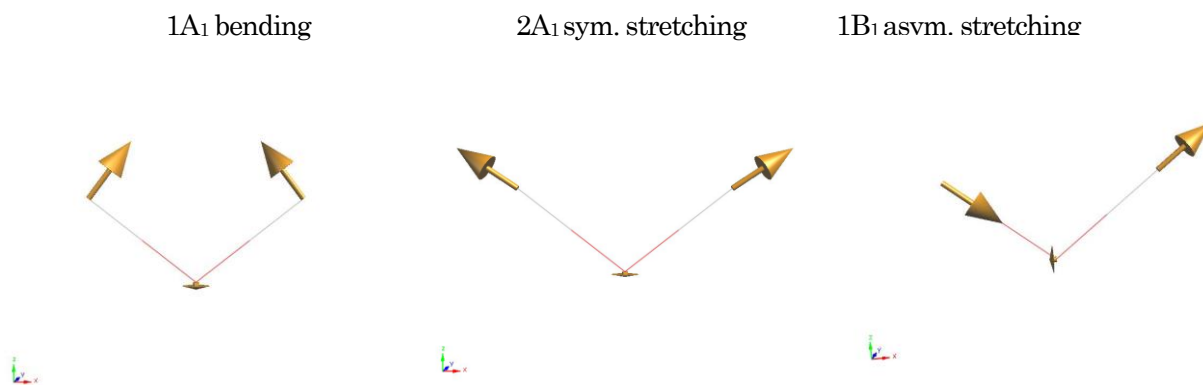


図 5-24 水分子の振動モードの固有ベクトル

5.2.1.5 計算例：シリコン結晶(Si2)

(1) 入力パラメータ

シリコン結晶の振動解析の例題です。計算例題は、sample/phonon/Si2 です。

入力パラメータファイル `nfinput.data` では、`element_list` にシリコン原子の質量 28.0855 amu が指定されています。質量の単位を原子質量単位とするため、`#units` の後に `atomic_mass` を指定しています

```
element_list{      #units atomic_mass
                  #tag element  atomicnumber mass
                  Si           14      28.0855
}
```

振動解析のパラメータを **Phonon** ブロックで指定します。

```
Phonon{
  sw_phonon = on
  sw_calc_force = on
  displacement = 0.1
  sw_vibrational_modes = on
}
```

`sw_calc_force` と `sw_vibrational_modes` がともに ON なので、振動解析のための力計算を行い、振動解析が行われます。

PHASE を実行します。

```
% mpirun ../.../bin/phase
```

計算が終了すると、出力ファイル `mode.data` に振動解析の結果が出力されます。`mode.data` の最初の部分は以下のようになっています。

```
--- primitive lattice vectors ---
0.0000000000  5.0875600000  5.0875600000
5.0875600000  0.0000000000  5.0875600000
5.0875600000  5.0875600000  0.0000000000
--- Equilibrium position and mass of each atom---
Natom=      2
  1  1.2718900000  1.2718900000  1.2718900000  51196.42133 Si
  2 -1.2718900000 -1.2718900000 -1.2718900000  51196.42133 Si
--- Vibrational modes ---
Nmode=      6 Natom=      2
n=      1 Tlu
  hbarW=  0.000000000E+00 Ha =  0.000000000E+00 eV; nu=  0.000000000E+00 cm^-1
  1  0.0000000000  0.0000000000  0.7071067812
  2  0.0000000000  0.0000000000  0.7071067812
```

最初の二行目から三行目は基本並進ベクトルをあらわしています。六行目は原子数を表しています。その次の行からは、原子の番号、デカルト座標、質量、ラベルが一行にあらわされています。**Vibrational modes** というタイトル行の次の行にはモード数と原子数があらわされています。これ以降には各振動モードの既約表現を先頭行として、次行に振動数があらわされ、その次の行から固有ベクトルがあらわされています。固有ベクトルは原子の番号の後にその原子に帰属するベクトルの3成分があらわされています。

(2) 振動数レベル図

振動解析の出力ファイル `mode.data` の振動数のデータから振動数レベル図を作成します。 以下のように、ツール `freq.pl` を実行すると、Postscript 形式の振動数レベル図 `freq.eps` が出力されます。

```
% freq.pl mode.data
```

シリコン結晶の振動解析の振動レベル図を図 5-25 に示します。この図から振動数が 517 cm^{-1} であるモードがあることが分かります。このモードの既約表現は T_{2g} であるので、同じ振動数のモードが三重に縮重しています。 T_{2g} モードがラマン活性である場合、図中の規約表現の右側に R が表示されます。赤外活性である場合には IR と表示されます。

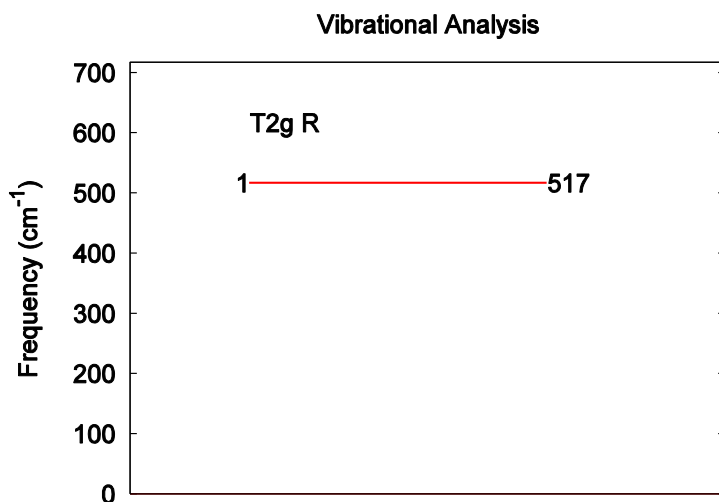


図 5-25 バルク Si の領域中心フォノンモードの振動数

(3) 振動モードの可視化

振動解析の出力ファイル `mode.data` から拡張 `Trajectory` 形式のファイルを作成することにより、固有ベクトルを矢印表示したり、原子が振動するアニメーションとして振動モードを可視化したりできます。ツール `animate.pl` を使用して、振動解析の出力ファイル `mode.data` から振動数の情報を取り出し、拡張 `Trajectory` 形式のファイル(拡張子:tr2)を作成します。

原点の移動とセルベクトルの変更を指定したファイル `control.inp` を用意します。`control.inp` は以下のように記述します。

```
origin 1.27189 1.27189 1.27189
vector1 10.17512 0 0
vector2 0 10.17512 0
vector3 0 0 10.17512
```

ツール `animate.pl` を以下のように実行すると、拡張 `Trajectory` 形式のファイルがモードの数だけ出力されます。

```
% animate.pl mode.data control.inp
```

この例題では切り出すセルをブラベー格子の単位胞にとり、セルの原点にシリコン原子がくるように設定しています。たとえば、出力された拡張 `Trajectory` 形式のファイル `mode_6.tr2` を可視化すると、図 5-26 のように固有ベクトルが矢印で示されます。図 5-26 に示されているセルは、出力された `grid.mol2` ファイルを読み込むことで表示できます。また、出力された拡張 `Trajectory` 形式から、原子の振動を可視化することができます。

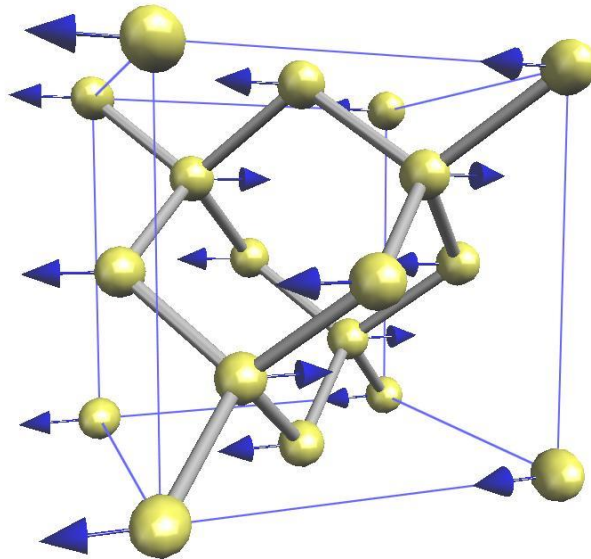


図 5-26 バルク Si の領域中心フォノンモードの固有ベクトル

5.2.2 フォノンバンド計算

5.2.2.1 機能の概要

PHASE には、 Γ 点だけでなく一般の k 点における格子振動解析を行い、フォノンの状態密度やバンド構造を計算する機能があります。

5.2.2.2 利用方法

(1) 基本的な入力パラメータ

この機能を利用するためには、 Γ 点の場合と同様 **phonon** ブロックを作成し、設定を行います。典型的には、以下ようになります。

```
phonon{
  sw_phonon = on
  sw_vibrational_modes = on
  sw_calc_force = on
  displacement = 0.1
  method = dos
  lattice{
    l1 = 2
    l2 = 2
    l3 = 2
  }
  dos{
    mesh{
      nx = 10
      ny = 10
      nz = 10
    }
  }
}
```

基本的な入力パラメーターを以下に示します。

第1 ブロック識別子	第2, 第3 ブロック識別子	タグ識別子	説明
phonon			フォノン計算の設定を行うためのブロック
		sw_phonon	PHASE の振動解析機能を利用するかどうかを指定します。 Γ 点のみの場合と同様です。
		sw_vibrational_modes	振動解析を行うかどうかを指定します。 Γ 点のみの場合と同様です。
		sw_calc_force	力定数の計算を行うかどうかを指定します。 Γ 点のみの場合と同様です。
		displacement	力定数の計算を行う際に、原子をどの程度移動させるかを指定します。 Γ 点のみの場合と同様です。デフォルト値は 0.1 bohr です。
		method	“手法”を指定します。状態密度計算の場合は dos, バンド構造計算の場合は band を指定します。
	lattice		一般 k 点の振動解析は、スーパーセル

			の力定数を必要とします。そのスーパーセルの大きさを指定するブロックです。
		nx	<i>a</i> 軸を何倍するかを指定します。
		ny	<i>b</i> 軸を何倍するかを指定します。
		nz	<i>c</i> 軸を何倍するかを指定します。
	dos		状態密度の計算方法を指定するブロックです。
	mesh		状態密度計算に利用する <i>k</i> 点のメッシュを指定するブロックです。
		nx	1 つめの逆格子ベクトルの分割数を指定します。
		ny	2 つめの逆格子ベクトルの分割数を指定します。
		nz	3 つめの逆格子ベクトルの分割数を指定します。

method を **band** と設定すると、フォノンバンドの計算になります。バンド構造の計算は、電子バンド構造と同様 **band_kpoint.pl** を利用して計算する *k* 点の情報が記録された **kpoint.data** ファイルを作成したあとに実行します。

(2) バンド数および *k* 点分割数

フォノンバンド計算を行う場合、スーパーセルの作成が行われます。バンド数や *k* 点分割数は、生成されたスーパーセルに合わせて **PHASE** が自動的に変更します。以下の注意が必要です。

- スーパーセルは、ブラベー格子に対して作成されます。通常の **PHASE** による計算の場合 **unit cell type** が **Bravais** の場合は基本格子に対して *k* 点を定義しますが、フォノンバンドの場合はブラベー格子に対して行うようにしてください。
- バンド数は、定義した原子にしたがって通常の計算と同じように決定してください。

(3) イオン性結晶の場合の設定方法

イオン性結晶の場合、 Γ 点においてオプティカルモードの縦波と横波が異なった振動数を持ちます。この現象は、**LO-TO** 分裂と呼ばれます。この効果を取り入れる場合、入力ファイルにおいてさらに以下の指定を行う必要があります。

```
phonon{
  sw_lo_to_splitting = on
  electronic_dielectric_constant{
    exx = 2.6
    eyy = 2.6
    ezz = 2.6
    exy = 0.0
    exz = 0.0
    eyz = 0.0
  }
}
```

変数 **sw_lo_to_splitting** を **on** とすることによって **LO-TO** 分裂を考慮した計算を行うことができます。**electronic_dielectric_constant** ブロックには、電子系の誘電テンソルを指定します。**electronic_dielectric_constant** ブロックの下に **exx**, **eyy**, **ezz**, **exy**, **exz**, **eyz** に対応する誘電テンソルの成分を指定します。誘電テンソルは、実測値もしくは **UVSOR-Epsilon** による計算値をご利用ください。原子の有効電荷も

指定する必要があります。これは、作業ディレクトリーに **effchg.data** ファイルを作成し、以下のように指定します。

```
2
1
1.12 0.0 0.0
0.0 1.12 0.0
0.0 0.0 1.12
2
-1.12 0.0 0.0
0.0 -1.12 0.0
0.0 0.0 -1.12
```

ファイルの 1 行目に原子数を記述します。2 行目以降に有効電荷の値を指定します。まず指定対象の原子の ID を指定し、さらに有効電荷テンソルを指定します。有効電荷テンソルは、形式電荷を利用することもできますが、UVSOR-Berry によって得られたボルン有効電荷を利用することが望ましいです。

(4) 計算の実行

入力データが準備できたら、通常通り **PHASE** を実行します。まずは、**PHASE** は入力の指定にしたがってスーパーセルを作成します。ログファイルには以下のように報告されます。

```
natm_super,natm2_super= 64 64
ia,cps(3),pos(3),ityp
1 1.27189 1.27189 1.27189 0.06250 0.06250 0.06250 1
2 8.90323 8.90323 8.90323 0.43750 0.43750 0.43750 1
3 1.27189 6.35945 6.35945 0.06250 0.31250 0.31250 1
4 8.90323 13.99079 13.99079 0.43750 0.68750 0.68750 1
5 6.35945 1.27189 6.35945 0.31250 0.06250 0.31250 1
6 13.99079 8.90323 13.99079 0.68750 0.43750 0.68750 1
7 6.35945 6.35945 1.27189 0.31250 0.31250 0.06250 1
8 13.99079 13.99079 8.90323 0.68750 0.68750 0.43750 1
9 11.44701 1.27189 1.27189 0.56250 0.06250 0.06250 1
10 19.07835 8.90323 8.90323 0.93750 0.43750 0.43750 1
11 11.44701 6.35945 6.35945 0.56250 0.31250 0.31250 1
12 19.07835 13.99079 13.99079 0.93750 0.68750 0.68750 1
13 16.53457 1.27189 6.35945 0.81250 0.06250 0.31250 1
...
...
```

natm_super がスーパーセルの原子数です。**cps** は原子のカルテシアン座標、**pos** はフラクショナル座標です。**ityp** は原子の種類を識別する番号です。また、スーパーセルに合わせて変化したバンド数と **k** 点のメッシュが次のように報告されます。

```
num_bands will be changed.
neg,meg= 192 192
k-point mesh will be changed.
mesh= 1 1 1
```

neg が新しいバンド数、**mesh** が新しい **k** 点メッシュです。

(5) 出力ファイル

mode.data ファイル

振動解析の結果は mode.data ファイルに記録されます。フォノンバンドの場合の mode.data ファイルは、たとえば以下ようになります。Γ 点の場合と比較して、振動モードの記述の仕方が異なります。

```
--- Vibrational modes ---
Nmode= 6 Natom= 2 Nqvec 120
iq= 1 q=( 0.00000, 0.00000, 0.00000) ( 0.00000, 0.00000, 0.00000)
n= 1 Tlu IR
hbarW= 0.000000000E+00 Ha = 0.000000000E+00 eV; nu= 0.000000000E+00 cm^-1
1 0.0000000000 0.7071067812 0.0000000000
2 0.0000000000 0.7071067812 0.0000000000
1 0.0000000000 0.0000000000 0.0000000000
2 0.0000000000 0.0000000000 0.0000000000
n= 2 Tlu IR
...
iq= 2 q=( 0.01875, 0.01875, 0.03750) ( 0.02316, 0.02316, 0.00000)
n= 1 B2 IR&R
hbarW= 0.63506708E-04 Ha = 0.17281054E-02 eV; nu= 0.13938112E+02 cm^-1
1 0.4999599615 -0.4999599615 0.0000000000
2 0.4999599615 -0.4999599615 0.0000000000
1 0.0063274755 -0.0063274755 0.0000000000
2 0.0063274755 -0.0063274755 0.0000000000
n= 2 B1 IR&R
...

```

モードの数と原子数の後に、 k 点の数が示されます。各 k 点の振動数モードの記述の前に、 k 点の座標が部分座標とカルテシアン座標で示されます。振動モードの固有ベクトルは一般には複素数となるので、固有ベクトルの実部の後に、虚部が記述されます。なお、Γ 点の場合と同様に振動の対称性および赤外ラマンの活性/不活性の判定が出力されますが、この情報は Γ 点以外では意味がない点にご注意ください。

phdos.data ファイル

フォノンの状態密度は phdos.data ファイルに出力されます。その内容は、典型的には下記のようなものです。

#	Index	Omega (mHa)	Omega (eV)	Omega (cm-1)	DOS (States/Ha)	DOS (States/eV)	DOS (States/cm-1)
IntDOS (States)							
0	-0.00050000	-0.00001361	-0.10973732	0.00000000	0.00000000	0.00000000	0.00000000
1	0.00950000	0.00025851	2.08500903	0.00473815	0.17412390	0.00002159	0.00001500
2	0.01950000	0.00053062	4.27975539	0.01996324	0.73363561	0.00009096	0.00012976
3	0.02950000	0.00080274	6.47450174	0.04568839	1.67901746	0.00020817	0.00044927
4	0.03950000	0.00107485	8.66924810	0.08191360	3.01026946	0.00037323	0.00107853
5	0.04950000	0.00134696	10.86399446	0.24722290	9.08527497	0.00112643	0.00286860
6	0.05950000	0.00161908	13.05874081	0.37130693	13.64527929	0.00169180	0.00591423
7	0.06950000	0.00189119	15.25348717	0.49343689	18.13347292	0.00224826	0.01020273
8	0.07950000	0.00216331	17.44823352	0.67844022	24.93222060	0.00309120	0.01602478
.....							
.....							

1 列目は状態密度のインデックス、2, 3, 4 列目がそれぞれ mHa, eV, cm-1 単位のエネルギー、5, 6, 7 列目がそれぞれ states/Ha, states/eV, states/cm-1 単位での状態密度、8 列目が積算状態密度です。積算状態密度は、最も高エネルギーの状態においては原子数×3 になります。

(6) 解析用 Perl スクリプト

フォノンバンド計算の結果解析用の Perl スクリプトが PHASE には備わっています。以下の 3 種類の Perl スクリプトを利用して結果の解析を行うことができます。

phonon_dos.pl

フォノンの状態密度データから、「フォノン状態密度図」を作成する Perl スクリプトです。以下のように使用します。

```
% phonon_dos.pl phdos.data OPTIONS
```

phdos.data が, PHASE が出力するフォノン状態密度データです. 実行すると, phonon_dos.eps という EPS 形式の画像ファイルが作成されます. 下記のオプションを利用することができます.

--units=UNITS or -u UNITS	エネルギーの単位を指定します. mHa, meV, THz, cm ⁻¹ のいずれかです. デフォルト値は cm ⁻¹ です.
--width=WIDTH or -w WIDTH	作成される図の幅を指定します. デフォルト値は 1 です.
--erange=[emin:emax] or -e [emin:emax]	エネルギーの範囲を指定します.
--drange=DRANGE or -d DRANGE	状態密度の範囲を指定します.
--title=TITLE or -t TITLE	図のタイトルを指定します.
--font=FONT or -f FONT	グラフに利用するフォントサイズを指定します. デフォルト値は 18 です.
--keep or -k	中間のデータファイルを保持する場合, このオプションを有効にします.
--mono or -m	モノクログラフを描画したい場合にこのオプションを指定します.
--dinc=DINC	状態密度の目盛を指定します.
--einc=EINC	エネルギーの目盛を指定します.

phonon_band.pl

フォノンバンドのデータから「フォノンバンド図」を作成する Perl スクリプトです。以下のように使用します。

```
% phonon_band.pl mode.data OPTIONS
```

mode.data が, 振動解析の結果が記録されたファイルです. 実行すると, phonon_band.eps という EPS 形式の画像ファイルが作成されます. オプションには, 下記のようなものがあります.

--control=CONTROL or -c CONTROL	band_kpoint.pl ファイルの入力ファイルを指定します. デフォルト値は bandkpt.in です.
--ptype=PTYPE or -p PTYPE	グラフ描画に利用するプロット種を指定します. line を指定すると実線, circle を指定すると丸でフォノンバンドを描画します. デフォルト値は line です.
--units=UNITS or -u UNITS	エネルギーの単位を指定します. mHa, meV, THz, cm ⁻¹ のいずれかです. デフォルト値は cm ⁻¹ です.
--width=WIDTH or -w WIDTH	作成される図の幅を指定します. デフォルト値は 1 です.
--erange=[emin:emax] or -e [emin:emax]	エネルギーの範囲を指定します.
--title=TITLE or -t TITLE	図のタイトルを指定します.
--font=FONT or -f FONT	グラフに利用するフォントサイズを指定します. デフォルト値は 18 です.
--mono or -m	モノクログラフを描画したい場合にこのオプションを指定します.
--keep or -k	中間のデータファイルを保持する場合, このオプションを有効にします.
--einc=EINC	エネルギーの目盛を指定します.

phonon_energy.pl

フォノンの状態密度から、振動に由来する内部エネルギーとヘルムホルツの自由エネルギーや比熱を計算するスクリプトです。振動に由来するヘルムホルツの自由エネルギーに通常の DFT 計算で得られる全エネルギーを加えれば、有限温度における固体の自由エネルギーを計算することができ、温度誘起の相転移を解析することなども可能です。

波数 \mathbf{k} のフォノンのエネルギーは、その振動数 $\omega_{\mathbf{k}}$ を利用して $(1/2 + n)\hbar\omega_{\mathbf{k}}$ と記述することができます。分配関数は $Q_{\mathbf{k}} = \sum_n e^{-U_{kn}/k_B T}$ と記述されるので、フォノンのエネルギーを代入し整理すると以下の結果が得られます。

$$Q_{\mathbf{k}} = \frac{\exp\left[-\frac{\hbar\omega_{\mathbf{k}}}{2k_B T}\right]}{1 - \exp\left[-\frac{\hbar\omega_{\mathbf{k}}}{k_B T}\right]}.$$

ヘルムホルツの自由エネルギーは $F_{\text{vib}} = \sum_{\mathbf{k}} -k_B T \log Q_{\mathbf{k}}$ と記述できるので、以下のように記述されます。

$$F_{\text{vib}} = \sum_{\mathbf{k}} \left[\frac{\hbar\omega_{\mathbf{k}}}{2} + k_B T \log \left(1 - \exp\left[-\frac{\hbar\omega_{\mathbf{k}}}{k_B T}\right] \right) \right].$$

振動による平均の内部エネルギーは $U_{\text{vib}} = \frac{1}{Q_{\mathbf{k}}} \sum_n U_{kn} \exp\left(-\frac{U_{kn}}{k_B T}\right)$ と記述できるので、以下のように記述することができます。

$$U_{\text{vib}} = \sum_{\mathbf{k}} \left[\frac{\hbar\omega_{\mathbf{k}}}{2} + \frac{\hbar\omega_{\mathbf{k}}}{\exp\left[\frac{\hbar\omega_{\mathbf{k}}}{k_B T}\right] - 1} \right].$$

エントロピーは、 F_{vib} および U_{vib} から $(U_{\text{vib}} - F_{\text{vib}})/T$ と計算することができます。定積比熱は内部エネルギーの温度に関する偏微分で与えられるので、以下のように計算することができます。

$$C_v = \frac{\partial U_{\text{vib}}}{\partial T} = k_B \left[\frac{\frac{\hbar\omega_{\mathbf{k}}}{k_B T} \exp\left(\frac{\hbar\omega_{\mathbf{k}}}{2k_B T}\right)}{\exp\left(\frac{\hbar\omega_{\mathbf{k}}}{k_B T}\right) - 1} \right]^2.$$

phonon_energy.pl を利用すると、以上のような計算を実行することが可能です。以下のように利用します。

```
% phonon_energy.pl mode.data
```

この操作によって、以下の 3 つのファイルが作成されます。

phonon_energy.data ファイル 内部エネルギー、ヘルムホルツ自由エネルギー、エントロピー、比熱の計算結果が以下の形式で記録されているファイルです。

```
# T (K) Internal Energy (eV) Free energy (eV) Entropy (eV/K) Cv (kB/atom)
0 0.125434126153072 0.125434126153072 0 0
30 0.12552700746085 0.125409486111375 3.91737831580881e-06 0.0820122071540538
60 0.126828216477476 0.124936822438767 3.15232339784872e-05 0.435633166874193
90 0.130001095247047 0.123379006005857 7.35787693465625e-05 0.787404251770626
120 0.134948880737123 0.120473935403623 0.000120624544445835 1.12444793146534
.....
.....
.....
```

1 列目に温度が, 2 列目以降からは内部エネルギーとヘルムホルツ自由エネルギーが eV 単位で, エントロピーが eV/K 単位で, 原子あたりの比熱が k_B 単位で記述されます。

phonon_energy.eps ファイル 内部エネルギー, ヘルムホルツ自由エネルギー, エントロピーを温度の関数としてプロットした EPS ファイルです。

phonon_Cv.eps ファイル 比熱と温度の関係をプロットした EPS ファイルです。

Si 結晶の場合に得られる **phonon_energy.eps** および **phonon_Cv.eps** の例を示します。 **phonon_energy.pl** スクリプトは, フォノン状態密度計算を実行した際に得られる **mode.data** ファイルを利用する必要がある点にご注意ください。フォノンバンド計算を実行した際に得られる **mode.data** ファイルを利用すると, 以下のようなエラーが発生し途中で終了してしまいます。なお, 得られるエネルギーは, 入力で指定した原子数分となります。

```
% phonon_energy.pl mode.data
weight undefined for q-point no. 1 at /home/user/phase/bin/phonon_energy.pl line 131, <MD> line 4450.
```

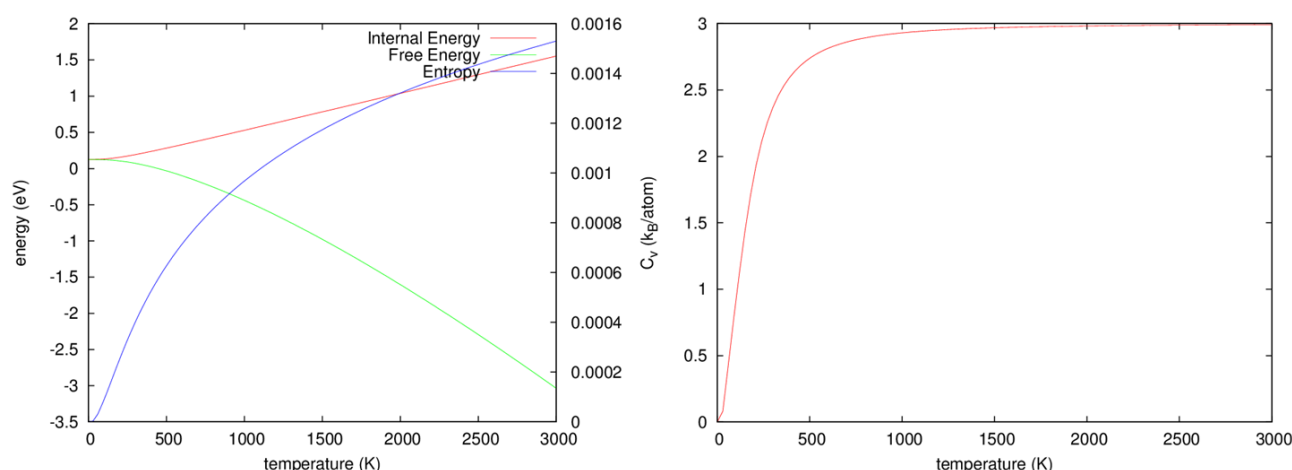


図 5-27 **phonon_energy.eps** (左図) と **phonon_Cv.eps** (右図)の例

phonon_energy.pl スクリプトのオプションは, 下記の通りです。

--width=WIDTH or -w WIDTH	作成される図の幅を指定します。デフォルト値は 1 です。
--trange=[tmin:tmax] or -t [tmin:tmax]	温度の範囲を指定します。デフォルト値は 0 K から 3000 K です。
--nT=NT or -n NT	温度の点の数を指定します。デフォルト値は 100 です。
--font=FONT or -f FONT	グラフに利用するフォントサイズを指定します。デフォルト値は 18 です。
--mono or -m	モノクログラフを描画したい場合にこのオプションを指定します。
--tinc=TINC	温度の目盛を指定します。
--einc=EINC	エネルギーの目盛を指定します。
--cinc=EINC	比熱の目盛を指定します。

5.2.2.3 例題

(1) シリコン結晶

最も簡単な例の 1 つとして, シリコン結晶のフォノンバンドとフォノン状態密度の計算を実行した例を紹介します。この例題の入力ファイルは, **samples/phband/Si** 以下にあります。

まずはバンド計算を行います。samples/phband/Si/band 以下の入力ファイルを利用します。
band_kpoint.pl 用の入力ファイル, bandkpt.in の内容は、以下のようになっています。

```
0.02
-0.83333333 0.83333333 0.83333333
0.83333333 -0.83333333 0.83333333
0.83333333 0.83333333 -0.83333333
0 0 0 1 # {/Symbol G}
1 1 0 2 # X
5 3 0 8 # U
0 0 0 1 # {/Symbol G}
1 0 0 2 # L
```

このbandkpt.in ファイルを利用して、以下のようにkpoint.data ファイルを作成します。

```
% band_kpoint.pl bandkpt.in
```

入力の、原子配置の指定は以下のようになっています。

```
structure{
  unit_cell_type = bravais
  unit_cell{
    a = 10.17512
    b = 10.17512
    c = 10.17512
    alpha = 90.0
    beta = 90.0
    gamma = 90.0
  }
  symmetry{
    tspace{
      lattice_system = facecentered
    }
    method = automatic
  }
  atom_list{
    coordinate_system = internal
    atoms{
      #tag element rx ry rz mobile
      Si 0.125 0.125 0.125 0
      Si 0.875 0.875 0.875 0
    }
  }
  element_list{
    #units atomic_mass
    #tag element atomicnumber mass
    Si 14 28.0855
  }
}
```

unit_cell_type をbravais とし, lattice_system パラメータによってこの系が facecentered, すなわち面心であることを指定しています。上述したように、通常の PHASE の計算ではこのような指定がなされている場合単位胞を基本格子に変換しますが、フォノンバンド計算ではそのようなことは行われません。次に、phonon ブロックを次のように記述しています。

```
Phonon{
  sw_phonon = on
}
```

```

sw_calc_force = on
sw_vibrational_modes = on
lattice{
  l1 = 2
  l2 = 2
  l3 = 2
}
method = band
}

```

スーパーセルは、 a, b, c 軸それぞれを 2 倍とする設定を採用しています。以上の設定のもと **PHASE** を通常通り実行します。計算が終了すると、その結果が mode.data に記録されます。mode.data ファイルからフォノンバンド図を得るためには、以下の操作を行います。

```
% phonon_band.pl mode.data --control=bandkpt.in
```

--control オプションで band kpoint.pl 用の入力ファイルを指定していますが、この指定がない場合はバンド図に特殊点を表す縦線などは描画されなくなります。結果は次に示す図のようになります。

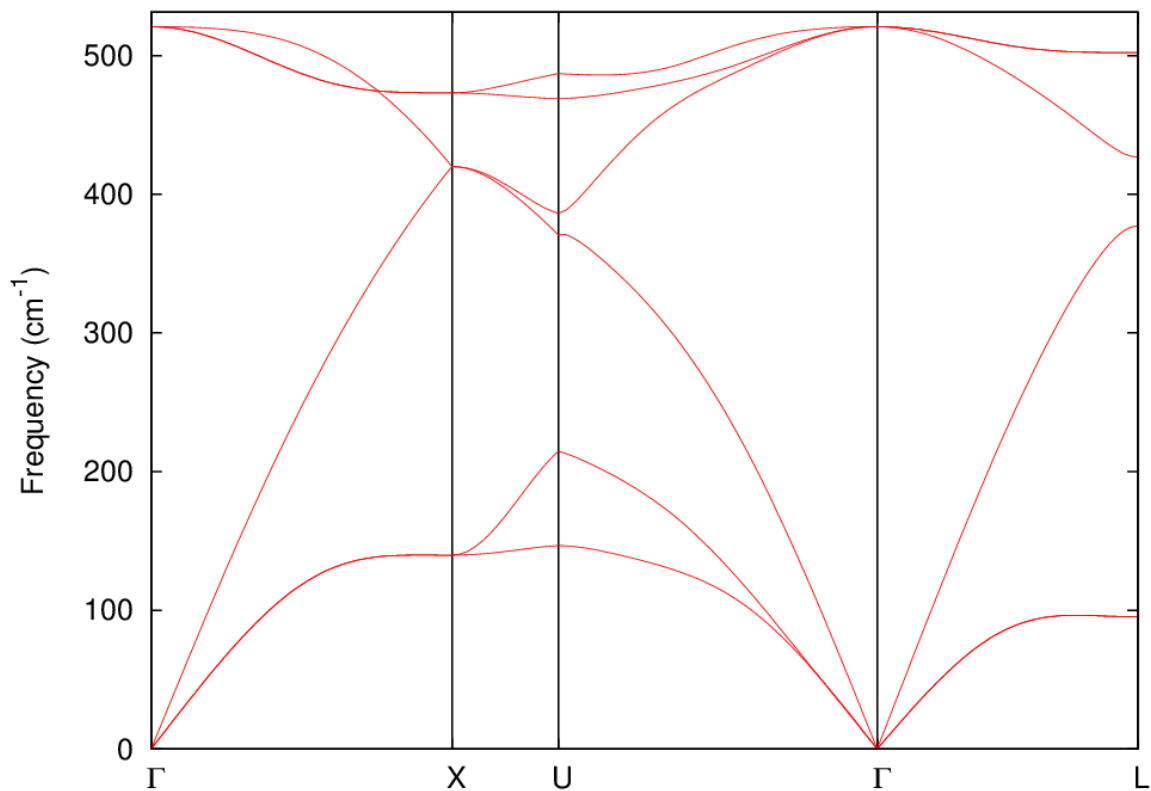


図 5-28 シリコン結晶のフォノンバンド

フォノンの状態密度の計算に必要な入力データは、samples/phband/Si/dos 以下にあります（力定数は計算済みなので、band ディレクトリーの下にある force.data ファイルをコピーして利用すると力定数計算をスキップすることも可能ですが、この場合は sw_calc_force パラメータを off に設定してください）このサンプルの入力パラメータファイルには、以下のような記述がなされています。

```

Phonon{
  sw_phonon = on
  sw_vibrational_modes = on
  lattice{
    l1 = 2

```

```

12 = 2
13 = 2
}
dos{
  mesh{
    nx = 10
    ny = 10
    nz = 10
  }
}
method = dos
}

```

method = dos と指定することによって状態密度計算を行うことを指定しています。dos ブロックの下に mesh ブロックにおいて、状態密度計算で利用するメッシュを $10 \times 10 \times 10$ としています。入力をこのように編集し終わったら PHASE を実行します。フォノン状態密度の計算結果は phdos.data ファイルに記録されます。このデータをもとに phonon dos.pl スクリプトを利用してフォノン状態密度図を作成します。

```
% phonon_dos.pl phdos.data
```

この結果得られるフォノン状態密度図は次に示す通りです。

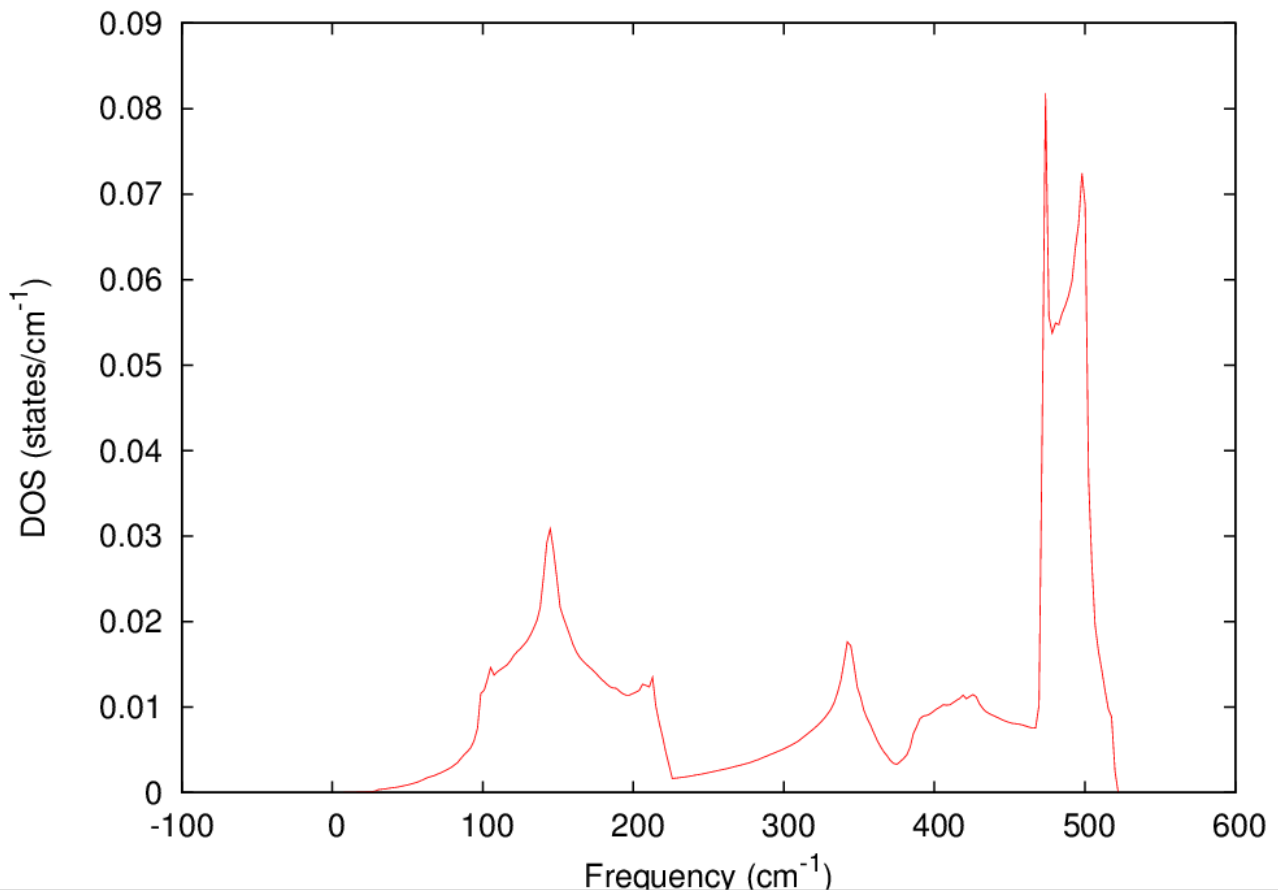


図 5-29 シリコン結晶のフォノン状態密度。

(2) ヨウ化カリウム

ヨウ化カリウムは NaCl 型の結晶構造をもつ、イオン性の結晶です。図 5-30 にその結晶構造を示します。ここでは、この結晶を例に LO-TO 分裂を考慮したフォノンバンド計算を紹介します。この例題の入力ファイルは、

samples/phband/KI 以下にあります。

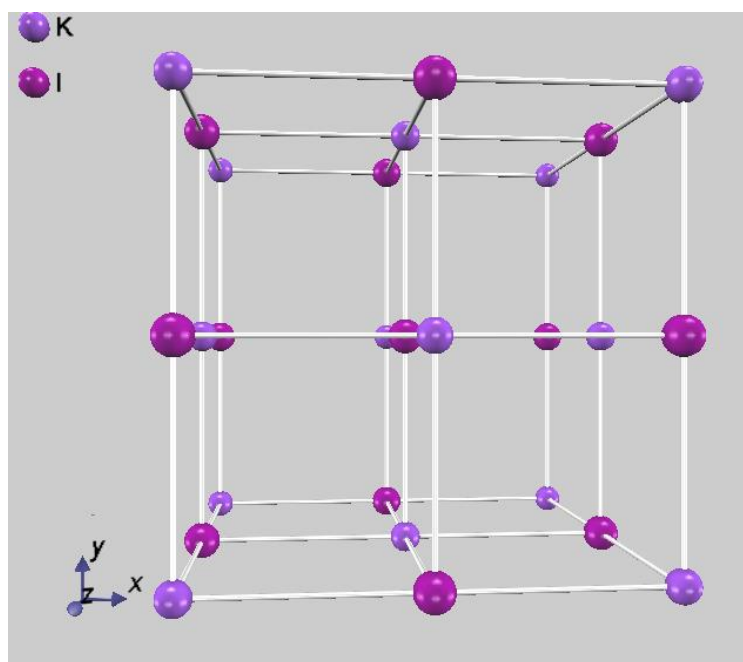


図 5-30 ヨウ化カリウムの結晶構造

LO-TO 分裂を考慮した計算を行うためには、電子系の誘電テンソルと有効電荷が必要です。これらは以下のようにして得ました。

- **誘電テンソル** : UVSOR-Epsilon を利用して計算しました。この際、2.2 eV のギャップ補正を施しました。結果は、xx, yy, zz 方向がそれぞれ 2.6 となりました。
- **有効電荷** : UVSOR-Berry を利用して、ボルン有効電荷テンソルの計算を行いました。結果は、カリウムの有効電荷が 1.1262, ヨウ素の有効電荷が -1.1262 となりました。

これらを設定し、sw_lo_to_splitting を on とする以外はシリコン結晶の場合と同じです。に、得られたフォノンバンドを示します。比較のため、LO-TO 分裂を考慮せずに計算した結果も合わせて表示しています。赤線が LO-TO 分裂を考慮せずに計算した結果、青線が考慮して計算した結果に対応します。この図から明らかなように、 Γ 点付近では LO-TO 分裂によって考慮しない場合は縮退している状態が分裂しています。

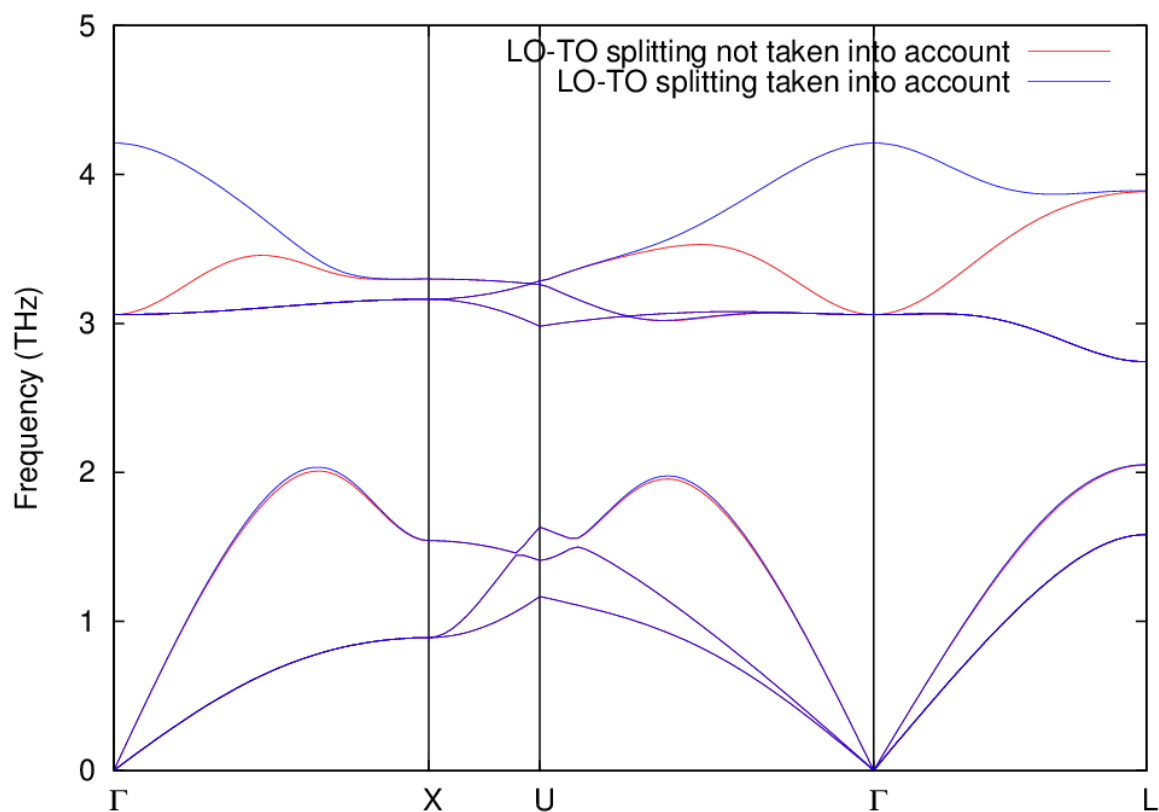


図 5-31 KI 結晶のフォノンバンド。赤線が LO-TO 分裂を考慮せずに計算した結果、青線が LO-TO 分裂を考慮して計算した結果。

(3) スズの温度誘起相転移

最後に、フォノン自由エネルギー解析の簡単な適用例としてスズの温度誘起相転移の例を紹介します。この例題の入力ファイルは、`samples/phband/Sn/a-Sn` (α スズ) および `samples/phband/Sn/b-Sn` (β スズ) にあります。

スズには、 α スズと β スズと呼ばれる同素体があります。 α スズはダイヤモンド構造、 β スズはその名の通り β スズ構造をとります。その結晶構造を、図 5-32 に示します。

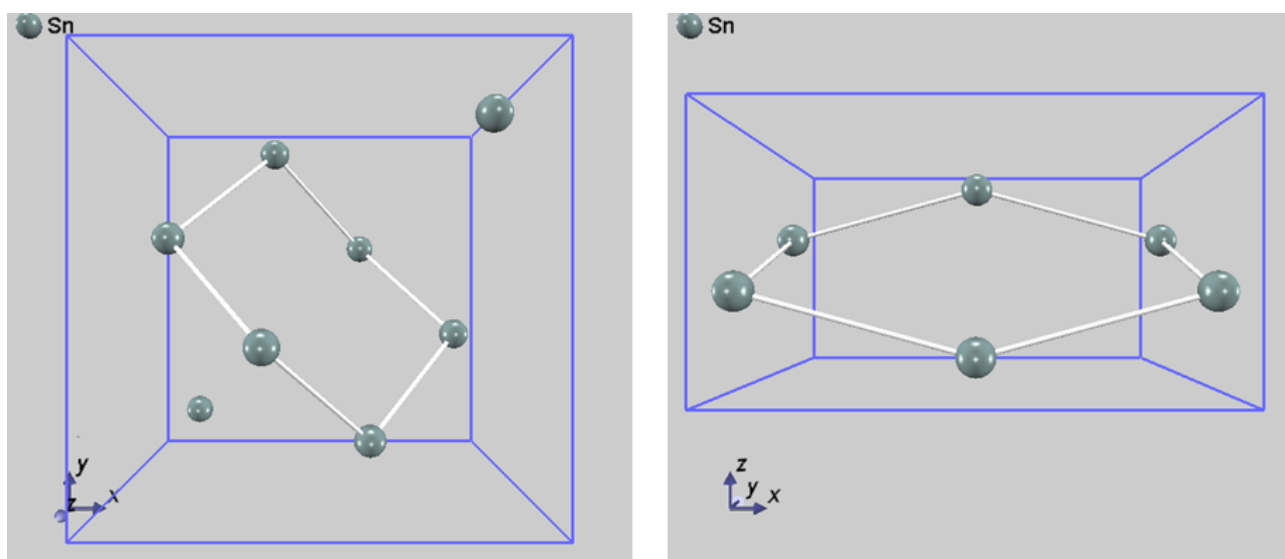


図 5-32 α スズ (左図) と β スズ (右図) の結晶構造

β スズ構造はダイヤモンド構造を c 軸方向から押しつぶしたような結晶構造であり、体心正方晶を取ります。常

温では β スズが安定ですが、低温下では α スズが安定になります。これは、結晶そのものの全エネルギーは α スズの方が低いですが、温度上昇に伴うフォノンの自由エネルギーの低下は β スズの方が大きいいためある温度で自由エネルギーは β スズの方が低くなり、相転移するからであると考えられます。このような現象を、フォノンの自由エネルギー計算と結晶の全エネルギー計算を組み合わせ確認していきます。

まずは、格子定数の最適化を行いました。ただし、 β スズ構造の ca 比は0.54614と固定して最適化しました。結果は、次の表に示す通りです。

	格子定数 a (Å)	格子定数 c (Å)	全エネルギー (ha/cell)
α スズ	6.6555	6.6555	-136.147884
β スズ	5.9184	3.2323	-136.144694

この結果から明らかなように、全エネルギーは α スズの方が低いので、絶対零度では α スズが安定であると考えられます。

続いて、得られた安定な格子定数のもとで振動解析を行いました。自由エネルギーを評価する場合に必要な計算は、状態密度のみです。 α スズ、 β スズに対してシリコンの場合と同様の設定を **Phonon** ブロックで行い、振動解析を実施しました。計算終了後に得られた mode.data ファイルを、phonon_energy.pl スクリプトで処理します。

```
% phonon_energy.pl mode.data
```

結果得られる phonon_energy.data ファイルの3列目にフォノンの自由エネルギーが記録されます。これは単位胞あたりの値なので、上述の全エネルギーの単位胞あたりのエネルギーを加え、温度の関数としてプロットすると図 5-33 のような結果が得られます。

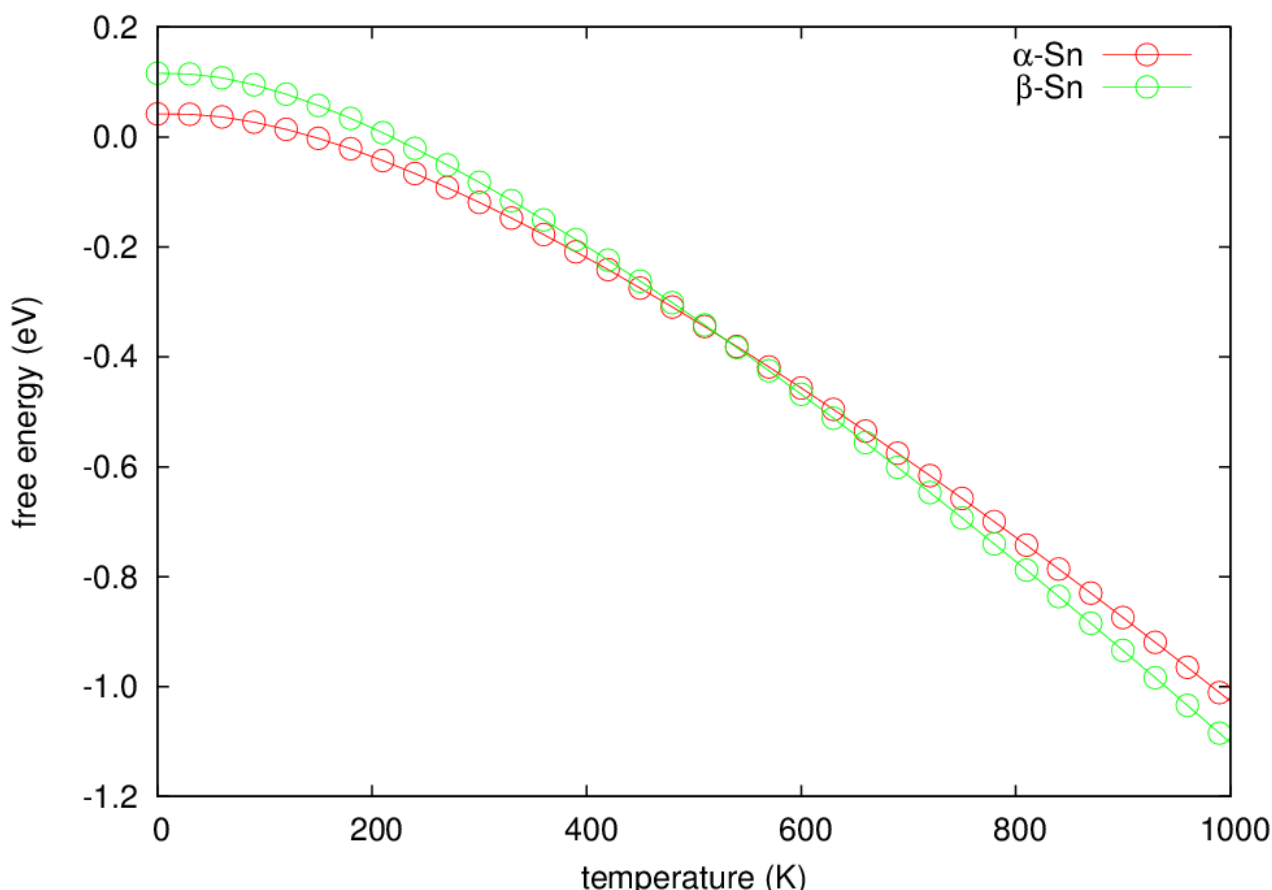


図 5-33 α スズと β スズの自由エネルギーと温度の関係。赤線が α スズ、緑線が β スズに対応する。

図 5-33 において α スズの曲線（赤線）と β スズの曲線（緑線）が交差する温度が転位温度と考えられます。この計算ではおおよそ 510 K となりました。実際には 290 K なので相転移温度が高く評価されてしまいましたが、このような計算によって温度誘起の構造相転移を説明できることはお分かりいただけたと思います。

5.2.2.4 使用上の注意

- 一般の \mathbf{k} 点における振動解析を実行するためには、スーパーセルに対する振動解析を行う必要があります。したがって、 Γ 点のみの場合と比較すると非常に多くの計算時間が必要です。
- フォノンバンド計算において最も計算量が多いのがスーパーセルに対する力の計算です。このデータは、1 度得られたら再利用することができます。たとえばフォノンの状態密度を計算したあとにフォノンバンドを計算する場合、または異なる対称線にそったフォノンバンドを計算する場合などは、以下のように `sw_calc_force` パラメータを `off` とすることによって力計算をやり直すことをさけることができます。

```
phonon{
    sw_phonon = on
    sw_calc_force = off
}
```

なお、力計算の結果が保存されているファイルは `force.data` というファイルです。バンド計算と状態密度計算を異なるディレクトリで行う場合に `sw_calc_force = off` とする場合はこのファイルを当該ディレクトリにコピーして利用してください。

- 通常の計算の場合、以下の指定を行うと、入力で指定された **Bravais** 格子は **Primitive** 格子に変換されて計算が行われます。

```
structure{
    unit_cell_type = bravais
    ...
    ...
    tspace{
        lattice_system = facecentered
    }
    ...
    ...
}
```

ところが、フォノンバンド計算の場合はこの変換は実施されず、**Bravais** 格子のままスーパーセルが作成され、計算が行われます。 \mathbf{k} 点サンプリングメッシュを検討する際などに注意が必要です。

- スーパーセル構築のパラメータ、`l1`, `l2`, `l3` はもとの対称性を保つような指定の仕方をしてください。異なる対称性の場合、意味のある計算は行われません。

5.2.3 分子動力学法シミュレーション

5.2.3.1 機能の概要

PHASE は、原子に働く力を利用して分子動力学法シミュレーションを行うことが可能です。 エネルギー一定、温度一定、圧力一定の分子動力学シミュレーションなどが実行できます。

5.2.3.2 入力パラメータ

分子動力学法シミュレーション機能と関連あるタグの一覧を表に示します。

分子動力学法シミュレーション機能に関連のあるタグの一覧

第1 ブロック識別子	第2, 第3 ブロック識別子	タグ識別子	説明
structure_evolution			原子座標データの更新方法を指定するブロック
		method	原子座標の更新方法を指定する。分子動力学シミュレーションの場合、 velocity_verlet (エネルギー一定の分子動力学シミュレーション) temperature_control (温度一定の分子動力学シミュレーション) pressure_control (エネルギー・圧力一定の分子動力学シミュレーション) temperature_pressure_control (温度・圧力一定の分子動力学シミュレーション)
		dt	時間刻みを指定する。 デフォルト値は 100 au (約 2.4 fs)
	temperature_control		温度制御の設定を行うブロック。
	method		温度制御の方法を指定する。nose_hoover か velocity_scaling のいずれか。 nose_hoover の場合は Nose-Hoover 熱浴による温度制御が、velocity_scaling の場合は温度スケールリングによる温度制御が行われる。デフォルト値は nose_hoover.
	sw_read_velocities		原子の初期速度を、PHASE/0 に自動生成させるのではなく手動で入力する場合にこのパラメータを on とします。デフォルト値は off です。
	set_initial_velocity		原子の初期速度をプログラムが自動的に設定するかどうかを指定するスイッチ。デフォルト値は on
	sw_shift_velocities		全運動量がゼロになるよう、MD ステップごとに速度をシフトするかどうかを指定するスイッチです。デフォルト値は off です。
	thermostat		熱浴の設定を行うブロック。表形式データ。
		temp	温度を指定する。
		qmass	熱浴の質量を指定する。
		tdamp	熱浴の質量を直接指定するのではなく、その周期を時間の単位で指定する。qmass に

			よる指定の方が優先される。なお、qmass も tdamp も指定がない場合、tdamp=50×dt に相当する周期の質量がデフォルト値として採用される。
	pressure_control		圧力制御の設定を行うブロック。
	pressure		目的の圧力を指定する。
	mass_baro		バーロスタットの質量を指定する。
	m11,m12,m13 m21,m22,m23 m31,m32,m33		格子の制御に拘束条件を加える。たとえば、m11 = off とすると 11 成分(a 軸・a 軸)が変化しなくなる。
structure	atom_list		
	atoms		原子配置を指定するブロック。表形式データ。
		mobile	原子が“可動”かどうかを指定する。可動にする場合 on を指定する。
		thermo group	原子に熱浴を割り当てる。定義した順に、整数値で指定する。 デフォルト値は 0 (熱浴に割り当てられていない状態)
		vx,vy,vz	原子の初期速度を手動入力する場合 (sw_read_velocities=on の場合)に各原子の速度の x,y,z 値を原子単位で入力する。入力が省略される場合、0 とみなされる。
	element_list		元素情報を指定する。表形式データ。
		mass	対応する元素の質量を指定する。デフォルト値は原子単位であることに注意。

5.2.3.3 計算結果の出力

座標データは file_names.data ファイルの F_DYNM によって指定されるファイルに各ステップでの座標値が出力されます。その形式は、構造最適化の場合と同様です。

● 原子座標

原子座標は、構造緩和を行った場合と同様、file_names.data 中の F_DYNM 識別子によって指定されるファイル (既定のファイル名は nfdynm.data)に記述されます。入力において printoutlevel ブロックの下の iprivelocity 変数の値を 2 以上にしていた場合 (分子動力学シミュレーションの場合はデフォルト値)、各原子の速度のデータも出力されます。速度のデータは、力のデータのあとに原子単位で出力されます。

● 各ステップでのエネルギー

各ステップでのエネルギーは、file_names.data 中の F_ENF 識別子によって指定されるファイル (既定のファイル名は nfeffn.data)に出力されます。サンプルによって得られる結果を以下に記します。

iter_ion	iter_total	etotal	ekina	econst	forcmx	
1	18	-7.8953179624	0.0000042358	-7.8953179624	0.0186964345	
2	30	-7.8953851218	0.0000665502	-7.8953185716	0.0183575424	
3	43	-7.8955768901	0.0002565396	-7.8953203505	0.0173392067	
4	56	-7.8958649874	0.0005418445	-7.8953231430	0.0156398790	
5	69	-7.8962052587	0.0008785990	-7.8953266596	0.0132645441	
6	83	-7.8965425397	0.0012120826	-7.8953304571	0.0102355854	
7	97	-7.8968179539	0.0014840140	-7.8953339398	0.0066063151	
8	111	-7.8969784478	0.0016420281	-7.8953364197	0.0024736141	
9	125	-7.8969875377	0.0016502900	-7.8953372478	0.0020111576	
10	139	-7.8968352058	0.0014992046	-7.8953360011	0.0066379641	
11	153	-7.8965440599	0.0012113794	-7.8953326806	0.0111430822	
.....						
.....						

一列目は原子座標の更新回数，二列目は電子の SCF 計算の回数です。三列目は，系の内部エネルギー，四列目は系の運動エネルギーです。五列目は系の内部エネルギーと運動エネルギーを足した値であり，エネルギー一定の分子動力学シミュレーションにおける保存量です。

5.2.3.4 使用方法：エネルギー一定の分子動力学シミュレーション

エネルギー一定の分子動力学シミュレーションの入力パラメータ例です。
計算例題は、samples/molecular_dynamics/NVE です。

```
accuracy{
  cutoff_wf = 9.00 rydberg
  cutoff_cd = 36.00 rydberg
  num_bands = 8
  xctype = ldapw91
  force_convergence{
    max_force = 1.0e-8 Hartree/Bohr
  }
  initial_wavefunctions = matrix_diagon
  ksampling{
    mesh{
      nx = 4
      ny = 4
      nz = 4
    }
  }
  scf_convergence{
    delta_total_energy = 1e-12 Hartree
    succession = 3
  }
}
...
...
structure{
  unit_cell_type = primitive
  unit_cell{
    a_vector = 0.0000000000      5.1300000000      5.1300000000
    b_vector = 5.1300000000      0.0000000000      5.1300000000
    c_vector = 5.1300000000      5.1300000000      0.0000000000
  }
  atom_list{
    atoms{
      #tag element rx ry rz mobile
      Si 0.130 0.130 0.130 yes
      Si -0.130 -0.130 -0.130 yes
    }
  }
  element_list{
    #tag element atomicnumber
    Si 14
  }
}
...
...
structure_evolution{
  method = velocity_verlet
```

```

    dt = 100
}
...

```

この入力、シリコン結晶の入力を少し変更したのとなっています。 **atoms** では、各原子の“mobile”変数を“yes”と設定しています。ここを“no”あるいは“0”と設定すると、その原子は 分子動力学シミュレーションを行っても動くことはありません。さらに座標値をあえて安定でない値にしています。具体的には、Si 結晶の二つの原子を(111)方向にお互いから離れるように少しだけずらしています。

(1) 手法の選択

structure_evolution ブロックでは、“method”変数を“velocity_verlet”としています。この選択によって 小正準集合の分子動力学シミュレーションを行うことができます。また、各ステップでの更新量(変数 **dt**)を、原子単位で“100”としています。上で述べたように、この値は $2.418 \times 10^{-15} \text{s}$ に相当します。

(2) 初期速度の与え方

ここまで説明したサンプルの入力を利用すると、原子の初期速度は全て 0 と設定されます。原子に初期速度を与える場合、下記のような入力を準備してください。

```

structure_evolution{
  method = velocity_verlet
  dt = 100
  temperature_control{
    thermostat{
      #tag temp
      300
    }
  }
}

```

ここで、“temp”変数で初期の温度をケルビン単位で設定します。原子の初期速度は、この温度になるように、かつ正規乱数に従って、全運動量が 0 になるように設定されます。

原子ごとに異なる初期温度を設定することも可能です。この場合、まず下記のような入力を作成します。

```

structure_evolution{
  method = velocity_verlet
  dt = 100
  temperature_control{
    thermostat{!#tag temp
      300
      500
      700
    }
  }
}

```

次に **atoms** の各原子に、“thermo_group”という変数を設定します。

```

structure{
  ...
  atom_list{
    atoms{
      !#tag rx ry rz element mobile weight thermo_group
      0.1159672611 0.1235205209 0.1215156388 Si 1 1 1
      -0.1329067626 -0.1264216714 -0.1225370484 Si 1 1 2
      0.1273740089 0.6305999369 0.6247606249 Si 1 1 3
      ...
    }
  }
}

```

```

    }
  }
  ...
}

```

この例では一番目の原子が 300K に、二番目の原子が 500K に、三番目の原子が 700K になるよう初期速度が設定されます。

(3) 計算結果の例

この計算例の計算結果の内部エネルギー，運動エネルギー，全エネルギーを図 5-34 に示します。

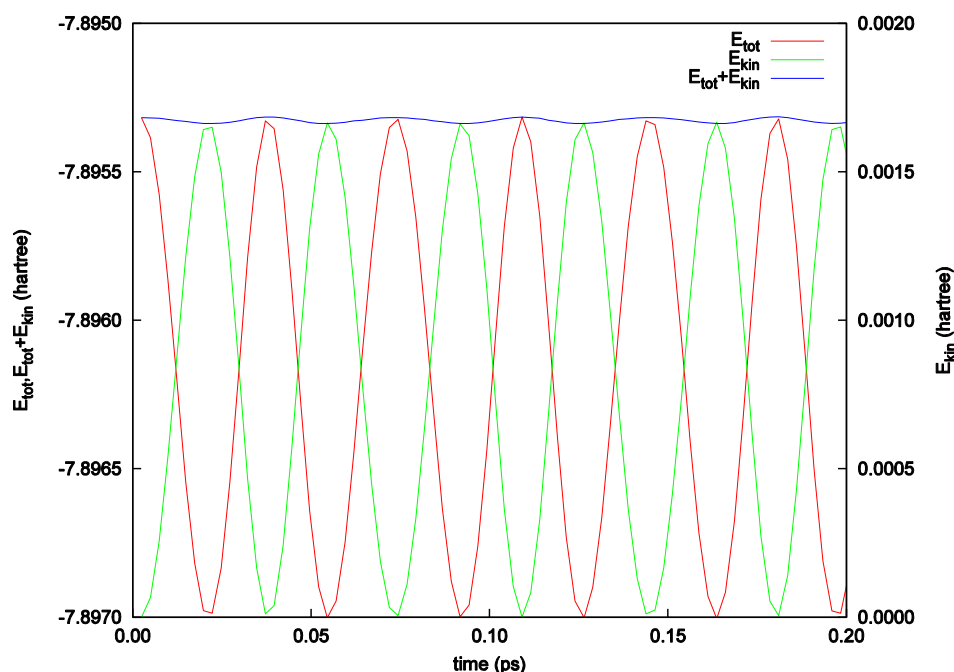


図 5-34 内部エネルギー，運動エネルギー，全エネルギーと時間の関係。

5.2.3.5 使用方法：温度一定の分子動力学シミュレーション

温度一定の分子動力学シミュレーションの入力パラメータ例です。

計算例題は、sample/molecular_dynamics/NVT です。

(1) 熱浴の設定

structure_evolution ブロックに temperature_control ブロックを指定します。

```

structure_evolution{
  method = temperature_control
  dt = 50.0
  temperature_control{
    thermostat{
      #tag temp qmass tdamp
      300 5000 10000
    }
  }
}

```

上記の入力例では、まず“method”変数を temperature_control としています。この変数によって温度制御を行うように指定します。ついで、“dt”変数を設定しています。これは、時間刻みの指定です。原子単位で入力します。例で示されている 50.0 という値は、約 1.2fs に相当します。

さらに、`temperature_control` ブロックで熱浴の設定を行っています。“`thermostat`”ブロックで各熱浴のパラメータを設定します。“`temp`”パラメータによってその熱浴の目的とする温度(ケルビン単位)，“`qmass`”パラメータによって熱浴の質量(原子単位)を設定します。“`qmass`”パラメータによって質量を直接指定するのではなく、“`tdamp`”パラメータ(時間の単位)によって熱浴の周期を指定し間接的に熱浴の質量を設定することも可能です。“`qmass`”と“`tdamp`”が両方設定されている場合，“`qmass`”が優先されます。また、いずれの指定もない場合 $50 \times dt$ の周期が実現するように熱浴の質量が設定されます。

(2) 熱浴の割り当て

`structure` ブロックの、`atoms` ブロックを設定する必要があります。設定例を以下に記します。

```
structure{
  ...
  atom_list{
    num_atoms = 8
    coordinate_system = internal
    atoms{
      !#tag rx ry rz element mobile weight thermo_group
      0.1159672611 0.1235205209 0.1215156388 Si 1 1 1
      -0.1329067626 -0.1264216714 -0.1225370484 Si 1 1 1
      0.1273740089 0.6305999369 0.6247606249 Si 1 1 1
      -0.1152089939 -0.6164829779 -0.6221565128 Si 1 1 1
      0.6299472943 0.1341313888 0.6253193197 Si 1 1 1
      -0.6305720382 -0.1290073650 -0.6187967685 Si 1 1 1
      0.6151271805 0.6206113965 0.1333834419 Si 1 1 1
      -0.6276524003 -0.6268549639 -0.1175099372 Si 1 1 1
    }
  }
  ...
}
```

各原子に“`thermo_group`”パラメータを割り振っています。このパラメータに 熱浴の識別番号を設定します。なお、熱浴の識別番号は熱浴の定義順に割り振られます。また、他の属性値と同様、“`#default`”タグを利用することによってデフォルト値を設定することも可能です。ここの例では全ての原子に同じ `thermo_group` を設定していますが、各原子が異なる熱浴に関連付けられていても問題ありません。

(3) 熱浴の“段数”の設定 (バージョン 2019.01 以上)

通常の熱浴をさらに熱浴で制御する、という計算手法が Nosé-Hoover chain 法です(Glenn J. Martyna and Michael L. Klein, and Mark Tuckerman, “Nosé-Hoover chains: The canonical ensemble via continuous dynamics”, *Journal of Chemical Physics* **97** 15 (1992)) 何段にもわたって再帰的に適用することも可能となっています。Nosé-Hoover chain 法を用いることによって、より少ないステップ数で熱浴が平衡状態に至ることなどが期待できます。利用するためには、以下のように `temperature_control` ブロックにおいて `num_chain` 変数を使います。

```
structure_evolution{
  method = temperature_control
  dt = 50.0
  temperature_control{
    num_chain = 5
    thermostat{
      #tag temp qmass tdamp
      300 5000 10000
    }
  }
}
```

理論的には各段数における熱浴の質量を別途設定することも可能ですが、最初の熱浴の質量以外結果にほとんど

影響を与えないため、のこりの熱浴の質量としては PHASE/0 が最初の熱浴の質量から自動的に算出した値が採用されるようになっています。

num_chain のデフォルト値は 1, すなわち通常の Nosé-Hoover 法です。

(4) “温度プロファイル” の設定 (バージョン 2019.01 以上)

シミュレーションの進行とともに入力パラメーターファイルの指定に応じてターゲット温度を変化させる分子動力学シミュレーションを行うことができます。変化のさせ方を“温度プロファイル”と呼びます。温度プロファイル機能を利用するためには、まず temperature_control ブロックにおいて sw_temperature_profile = on とします。さらに, thermostat ブロックにおいて温度プロファイルの設定を行います。温度プロファイル機能を利用していない場合 thermostat ブロックにおいて定義するテーブルの行は一つの熱浴を表しますが, 利用している場合はテーブルの行は一つのプロファイルを表すことに注意が必要です。温度プロファイル機能を利用している場合にさらに複数の熱浴を定義するためには, 属性値 no (id, thermo_group も可)によって切り分けます。具体的には, たとえば下記に示すようになります。

```
structure_evolution{
  ...
  ...
  temperature_control{
    sw_temperature_profile = on
    thermostat{
      #tag no   tempi tempf till_n tdamp
      1  8000  8000 3000  5000
      1  8000   300  7000  5000
      2   400   500  1000  5000
    }
  }
}
```

上述のように属性値 no によって熱浴を指定します。tempi, tempf はそれぞれ初期温度および終温度に対応する属性値です。till_n は, この温度プロファイルを採用する最大 MD ステップ数です。属性値 tdamp によって熱浴の緩和時間を指定します。tdamp のかわりに属性値 qmass を使って熱浴の質量を直接指定することも可能です。以上より, この例では thermostat テーブルの各行は次のように解釈されます。

- 1 行目: 1 つ目の熱浴の温度を 8000K で 3000 ステップ適用する。
- 2 行目: 1 つ目の熱浴の温度を 8000K から 300K まで 3001 ステップから 7000 ステップまで適用する。
- 3 行目: 2 つ目の熱浴の温度を 400K から 500K まで 1000 ステップまで適用する。

なお, 最後のプロファイルの till_n 以降は, そのプロファイルの tempf の温度が使い続けられる仕様になっています。

5.2.3.6 使用方法: 温度・圧力一定の分子動力学シミュレーション

PHASE/0 は, Ivo Souza and JoséLuís Martins, Phys. Rev. B 55 (1997) pp. 8733-8742, E. Hernández, Journal of Chemical Physics, 115 (2001) pp. 10282-10290 などの文献において解説されている手法によって圧力一定の分子動力学シミュレーションを行うことができます。

(1) 入力パラメーターファイルの書き方

まず, 温度一定の分子動力学シミュレーションと同様の手続きで入力パラメーターファイルを作成します。通常の温度一定の場合との違いは, 圧力も一定であること, また圧力浴の設定が必要なことの 2 点です。以下に例を示します。

```
structure_evolution{
  method = temperature_pressure_control
```



```

dt = 50.0
temperature_control{
  num_thermostat = 1
  set_initial_velocity = on
  !!method = velocity_scaling
  thermostat{
    #tag temp qmass
    300 4000
  }
}
pressure_control{
  pressure = 0.0
  mass_baro = 1
  m11 = on
  m22 = on
  m33 = on
  m12 = on
  m13 = on
  m23 = on
}
}

```

設定方法の詳細は、下記の通り。

- MD 手法の選び方

`method` に `temperature_pressure_control` もしくは `pressure_temperature_control` を指定すると温度-圧力一定の分子動力学シミュレーション、すなわち NPT 分子動力学シミュレーションを実行することができます。 `pressure_control` とすると圧力一定の分子動力学シミュレーション、すなわち NPH 分子動力学シミュレーションを実行することができます。

- 温度制御の設定

温度制御は、NVT の場合と同様 `temperature_control` ブロックにおいて行います。ただし、NVT シミュレーションでは複数の熱浴を定義することが可能であるのに対し、NPT シミュレーションの場合に定義できる熱浴は一つのみである点に注意してください。また、上記の例ではコメントアウトされていますが、`method = velocity_scaling` と指定することによって速度スケールリングによる温度制御を利用することもできます。Nosé-Poincaré の熱浴で温度制御を行うことが理想的ですが、安定に運動方程式が時間発展できない場合は速度スケールリングを利用してください。指定がなければ Nosé-Poincaré 熱浴が採用されます。

- 圧力制御の設定

圧力制御は、`pressure_control` ブロックを作成し、その下で設定します。`pressure` に目的の圧力を指定します。圧力の単位のデフォルト値は原子単位ですが、GPa などを利用することもできます。`mass_baro` には熱浴の質量を指定します。デフォルト値は 1 であり、単位の指定はできません。目安としては、熱浴の周期（単位胞の体積変化の周期）が 100 MD ステップ相当程度の時間になるように決めるとよいでしょう。

- 計量テンソルに対する拘束条件の設定

`mxy` パラメーターによって、計量テンソルの変化に対して拘束条件を課すことができます。たとえば `m11 = off` とすると 11 成分(a 軸・ a 軸)が変化しなくなり、`m12 = on` ならば 12 成分(a 軸・ b 軸)および 21 成分(b 軸・ a 軸)が変化しなくなります。このパラメーターのデフォルト値はすべて `on` なので、特に指定がなければ計量テンソルのすべての成分が変化します。

NPT シミュレーション結果で得られる主な出力ファイルは、エネルギーなどの履歴を記録した `nfefn.data` ファイル、座標データの履歴を記録した `nfdynm.data` ファイル、計量テンソルの履歴を記録した `nfmetric.data` ファイル、そして格子定数の履歴を記録した `nflatconst.data` ファイルである。それぞれについて説明する。

(2) `nfefn.data` ファイル

エネルギーなどの履歴を記録したファイルであり、構造最適化や NVT の MD シミュレーションの場合も得られ

ます。NPT シミュレーションの場合、以下のような出力が得られます。なお、紙幅の都合で 1 行を改行して表示しています。

iter_ion, iter_total, etotal, ekina, econst, pressure				
1	13	-31.8045273788	300.0000000000	0.0000000000
0.0209863336	-0.0000504658			
2	25	-31.8045248143	301.4440100456	0.0000586853
0.0211839341	-0.0000573770			
3	37	-31.8043935680	299.1271868912	0.0001085792
0.0214781523	-0.0000574428			
4	49	-31.8041402859	293.2700519011	0.0001476826
0.0218951630	-0.0000570036			
5	61	-31.8037768714	284.3267516440	0.0001733906
0.0224437768	-0.0000560637			
6	73	-31.8033194235	272.9264725662	0.0001832795
0.0231125156	-0.0000546420			
7	85	-31.8027870051	259.8057152864	0.0001752150
0.0238698884	-0.0000527691			
8	97	-31.8022003504	245.7426304561	0.0001475268
0.0246685850	-0.0000504870			
9	109	-31.8015806171	231.5028071591	0.0000992076
0.0270606007	-0.0000478460			
10	121	-31.8008921515	217.8018629938	0.0000862100
0.0294678421	-0.0000449028			
11	133	-31.8002640917	205.3149756742	0.0000004873
0.0315593847	-0.0000420615			
...				

...

一行が 1 タイムステップのデータに相当します。1 列目が原子（単位胞）の更新回数, 2 列目が SCF 計算の総更新回数, 3 列目が系のエネルギー（原子単位）, 4 列目が瞬間的な温度（ケルビン単位）, 5 列目がハミルトニアン（原子単位）, 6 列目が原子に働く力の最大値（原子単位）, 7 列目が瞬間的な圧力です（すべて原子単位）。ハミルトニアンは理想的には保存するはずですが, 第一原理計算においてエネルギーに対する歪みテンソルの厳密な微分を求めることは困難なため, 保存がよくない傾向にある点に注意してください。

(3) nfdynm.data ファイル

基本的なファイルフォーマットは, 通常の nfdynm.data ファイルと同じです。ただし, NPT シミュレーションの場合毎ステップ格子定数が増加するので, 格子定数などが記述されるヘッダー部が毎ステップ書き込まれる点に違いがあります。

(4) nfmetric.data ファイル

各ステップにおける計量テンソルが記録されるファイルです。典型的な内容は下記の通り。

2		
105.8775163849	0.0064685429	-0.0017420207
0.0064685429	105.8774556260	0.0027757249
-0.0017420207	0.0027757249	105.8776712432
3		
105.8648537936	0.0196786348	-0.0055233812
0.0196786348	105.8644796107	0.0091955626
-0.0055233812	0.0091955626	105.8652454083
4		
105.8462013539	0.0389223098	-0.0123434716
0.0389223098	105.8452005929	0.0190717120
-0.0123434716	0.0190717120	105.8468474757

まずステップ数を表わす整数値が記録され, ついで 3×3 のテンソルが記録されます。なお, 半ステップずれた数値解法を採用しているため, 記録は 2 ステップ目からとなります。

(5) nflatconst.data ファイル

各ステップにおける格子定数が記録されるファイルです。典型的な内容は下記の通り。なお, 紙幅の都合で 1 行を改行して表示している

2	10.2896800915	10.2896771391	10.2896876164	89.9984979129
90.0009426965	89.9964995371	1089.4462540511		
3	10.2890647677	10.2890465841	10.2890837983	89.9950232125
90.0029893382	89.9893495841	1089.2504025643		
4	10.2881583072	10.2881096705	10.2881897084	89.9896762432
90.0066816443	89.9789308011	1088.9605527024		
5	10.2869678503	10.2868710406	10.2870094050	89.9825817981
90.0125180935	89.9656805224	1088.5783846407		
6	10.2855038285	10.2853392827	10.2855504459	89.9738915620
90.0209385359	89.9500901373	1088.1067192172		
7	10.2837804786	10.2835276570	10.2838245727	89.9637869404
90.0323061920	89.9327006608	1087.5497224800		
8	10.2818163114	10.2814547573	10.2818483397	89.9524812878
90.0468878988	89.9140944258	1086.9130916478		
9	10.2796344063	10.2791449593	10.2796435437	89.9402204612

一行が 1 タイムステップのデータに相当します。1 列目は単位胞の更新回数, 2 列目から 7 列目がそれぞれ格子定数 $a, b, c, \alpha, \beta, \gamma$, 最後の 8 列目が単位胞の体積です。長さの単位は Bohr 単位, 角度の単位は度, 体積の単位は Bohr³ です。

5.2.3.7 分子動力学シミュレーションにまつわるその他の設定

(1) 速度スケールリングによる温度制御の分子動力学シミュレーション

Nosé-Hoover の熱浴ではなく, 原子の速度を温度が合うようにスケールし直すことによって温度を制御することも可能です。このような計算は, `temperature_control` ブロックの下に `method` 変数を `velocity_scaling` とすることによって実現することができます。

```
structure_evolution{
  ...
  temperature_control{
    method = velocity_scaling
    ...
  }
}
```

速度は, 目的の温度に合うよう毎 MD ステップスケールされます。

(2) 全運動量がゼロになるよう速度をシフトする方法

分子動力学シミュレーションにおいて理論上全運動量は保存します (ゼロになる) が, 実際は数値誤差によりゼロとはならず, 系全体が並進してしまう場合があります。これを防ぐには, 以下のように変数 `sw_shift_velocities` の値を `on` とし, MD 計算中全運動量がゼロとなるようにします。

```
structure_evolution{
  ...
  temperature_control{
    ...
    sw_shift_velocities = on
  }
}
```

(3) 原子の初期速度を手動で指定する方法

原子の初期速度は通常ランダムな正規分布が得られ, 対応する温度が設定した温度で, 全運動量が 0 になるように自動的に決定されますが, 手動で各原子に割り振ることも可能です。このような指定は, 原子配置の `vx, vy, vz` 属性値によって行います。また, 設定した速度を上書きしてしまわないよう `temperature_control` ブロックの下において `sw_read_velocities = on` を設定する必要があります。たとえば, 以下のように設定します。

```
structure{
  ...
  atom_list{
    atoms{
      !#tag rx ry rz element mobile thermo_group vx vy vz
      0.11 0.12 0.11 Si 1 1 0.001 0.0014 0.0008
      -0.13 -0.13 -0.14 Si 1 1 -0.001 -0.002 0.0001
      0.12 0.63 0.62 Si 1 1 0.0003 -0.0005 -0.00028
      ...
    }
  }
}
```

```

        ...
    }
}
...
structure_evolution{
    ...
    temperature_control{
        sw_read_velocities = on
    }
}

```

なお、速度の単位のリファレンス値は原子単位です。

また、この方法で初期速度を指定した場合でも、デフォルトの設定では温度に合うよう、また全運動量が 0 になるよう速度はソフト・スケールされます。この動作を抑制したい場合、以下の要領で変数 `set_initial_velocity` の値を `off` とします。

```

structure_evolution{
    ...
    temperature_control{
        sw_read_velocities = on
        set_initial_velocity = off
    }
}

```

(4) 原子を領域に閉じ込める方法 (バージョン 2019.01 以上)

矩形領域および円筒領域に閉じ込めることができます。このような計算機能は、狭い領域に閉じ込められた系の振る舞いを調べたり、ESM 法や dipole 補正法などアルゴリズムの都合上真空層が必須である機能において真空層に原子が入ってこられなくする目的で利用することができます。

入力の記述

領域は、`structure` ブロックの `regionx` ブロックにおいて定義することができます。ここで `x` は領域の識別番号です。

```

structure
  region1{
    region_group = 1
    type = cylinder
    radius = 3.5 angstrom !半径
    cylx = 5 angstrom
    cyly = 5 angstrom ! 円筒の中心位置
    orientation = 3 ! 円筒の向き
    cylzmin = -1000 ! 円筒の下端
    cylzmax = 1000 ! 円筒の上端
    sw_tally = on ! エネルギーを加える
    eps = 0.001 !ポテンシャルの深さ
    sigma = 1.5 !距離のスケール
  }

```

```

}

```

x は 1 から始まります。 x を 2, 3, ... とすることによって任意の数の領域を定義することができます。`regionx` ブロックにおいて以下の変数を設定することによって領域を定義します。

変数名	説明
<code>region_group</code>	“領域のグループ” を整数で指定します。この値が同じ領域はひとかたまりの “グループ” として扱われます。後述の原子配置テーブルの属性値 <code>region_group</code> にこの数値を指定することによって領域グループと原子を紐づけます。デフォルト値は <code>regionx</code> ブロックの x の値です。
<code>type</code>	領域の種類を指定します。 <code>cylinder</code> (円筒型) もしくは <code>box</code> (直方体型) とします。デフォルト値は <code>box</code> です。
<code>radius</code>	円筒の半径を指定します。デフォルト値は 6.5 bohr です。 <code>type=cylinder</code> の場合のみ意味のある設定です。
<code>cylx</code>	円筒の中心位置の x 座標を指定します。デフォルト値はセルの境界です。 <code>type=cylinder</code> の場合のみ意味のある設定です。
<code>cylx</code>	円筒の中心位置の y 座標を指定します。デフォルト値はセルの境界です。 <code>type=cylinder</code> の場合のみ意味のある設定です。
<code>orientation</code>	円筒の向きを指定します。1 の場合 x 方向, 2 の場合 y 方向, 3 の場合 z 方向に向きます。デフォルト値は 3 です。 <code>type=cylinder</code> の場合のみ意味のある設定です。
<code>cylzmin</code>	円筒の長さ方向の下限を指定します。デフォルト値は -10^{10} です。 <code>type=cylinder</code> の場合のみ意味のある設定です。
<code>cylzmax</code>	円筒の長さ方向の上限を指定します。デフォルト値は 10^{10} です。 <code>type=cylinder</code> の場合のみ意味のある設定です。
<code>xmin</code>	直方体の x 方向の下限値を指定します。デフォルト値は -10^{10} です。 <code>type=box</code> の場合のみ意味のある設定です。
<code>xmax</code>	直方体の x 方向の上限値を指定します。デフォルト値は 10^{10} です。 <code>type=box</code> の場合のみ意味のある設定です。
<code>ymin</code>	直方体の y 方向の下限値を指定します。デフォルト値は -10^{10} です。 <code>type=box</code> の場合のみ意味のある設定です。
<code>ymax</code>	直方体の y 方向の上限値を指定します。デフォルト値は 10^{10} です。 <code>type=box</code> の場合のみ意味のある設定です。
<code>zmin</code>	直方体の z 方向の下限値を指定します。デフォルト値は -10^{10} です。 <code>type=box</code> の場合のみ意味のある設定です。
<code>zmax</code>	直方体の z 方向の上限値を指定します。デフォルト値は 10^{10} です。 <code>type=box</code> の場合のみ意味のある設定です。
<code>sw_tally</code>	領域に起因するエネルギーを全エネルギーに加える場合この値を <code>on</code> に設定します。
<code>eps</code>	ポテンシャル $\epsilon \left(\frac{\sigma}{r} \right)^{12}$ の ϵ の値をエネルギーの単位で指定します。デフォルト値は 1e-3 hartree です。
<code>sigma</code>	ポテンシャル $\epsilon \left(\frac{\sigma}{r} \right)^{12}$ の σ の値を長さの単位で指定します。デフォルト値は 1 bohr です。

定義した領域は、その `region_id` を紐づけたい原子の `region_id` に指定することによって割り当てることができます。たとえば、以下のように設定します。

```

structure{
  ...
  atom_list{

```

```

coordinate_system = cartesian
atoms{
  #default mobile=on,thermo_group=1
  #units angstrom
  #tag    element    rx      ry      rz region_group
  H      4.231707      4.904619      6.374683 1
  H      5.716594      4.994127      6.011627 1
  O      5.118193      4.883964      6.766158 1
  H      4.167342      5.876768      8.210465 2
  H      5.481543      5.259672      8.697061 2
  O      4.627457      5.590603      9.014168 2
  ...
  ...
}
}
}

```

この例では、1 番目、2 番目、3 番目の原子は **region_group=1** のすべての領域に紐づけられ、4 番目、5 番目、6 番目の原子は **region_group=2** のすべての領域に紐づけられます。

計算の実行

計算は、通常の分子動力学シミュレーションと同じように実行することができます。入力パラメーターファイルが読み込まれると以下のように読み込んだ領域の情報が報告されます。

```

!** region statistics
!** num_regions =      1
!** status for region no      1
!** region type      : CYLINDER
!** orientation      :   3 (1->x, 2->y, 3->z)
!** radius           :      6.6140409725
!** cylx,cyly        :      9.4486299607      9.4486299607
!**cylzmin,cylzmax: -1.797693134862316E+308 1.797693134862316E+308
!** sigma, epsilon   :      1.00000000000      0.00100000000
!** tally            : F
!** n target atoms   :      36
   1      2      3      4      5      6      7      8
   9     10     11     12     13     14     15     16
  17     18     19     20     21     22     23     24
  25     26     27     28     29     30     31     32
  33     37     38     39

```

特に、n target atoms 以降領域に割り当てられている原子のリストが出力されるので、想定通り領域が設定できているかどうかを確認することができます。

5.2.3.8 使用における注意点

- 分子動力学シミュレーション機能に特別な制約はありません。ウルトラソフトおよび PAW 擬ポテンシャル、並列計算、継続計算に対応しています。ただし、分子動力学シミュレーションにおいては原子の位置は熱揺らぎによって激しく振動するので、系が厳密な意味での対称性を持つことはありません。したがって、対

称性関連の設定は施さないでください。

- 原子の質量は、通常の構造最適化においては直接系の性質を左右するものではありませんが、分子動力学シミュレーションの場合は意味のある量です。したがって、本機能を利用する場合は元素の正しい質量を指定する必要があります。PHASEにおける標準の質量の単位は、原子単位です。たとえば、陽子の質量は原子単位で1822.877333 です。
- 運動エネルギーはハートリー単位で記述されますが、運動エネルギーと温度との間には $E_{\text{kin}} = \frac{3}{2} \times N_{\text{atom}} \times k_{\text{B}}T$ という関係があります。ここで運動エネルギーを E_{kin} 、原子数を N_{atom} 、ボルツマン定数を k_{B} 、瞬間的な温度を T と記述しました。従って運動エネルギーから系の温度を知りたい場合、まず運動エネルギーを原子数で割り、 3.1578×10^5 という値を掛けて(ハートリー単位から $k_{\text{B}}T$ 単位への変換)、さらに $\frac{3}{2}$ で割って下さい。
- 原子座標の更新回数は入力で指定した時間刻みの値(structure_evolution ブロック内の dt という変数で指定)を掛けること によって実時間での経過が分かります。時間の単位は入力で指定することが可能ですが、原子単位系を利用した場合(デフォルト) 2.418×10^{-17} という値を掛ければ「秒」に変換することが可能です。たとえば 100 a.u. という時間は 2.418 fs に相当します。
- 温度一定の分子動力学シミュレーションにおける、熱浴の質量 Q の値について注意点を挙げます。小さすぎる Q を採用すると、系のダイナミクスに熱浴に起因する 人為的なモードが生じてしまい、また場合によっては計算が破綻してしまいます。他方大きすぎる Q を採用すると、系が熱平衡に達するのに 多くのステップ数を必要とするようになってしまいます。

Q の値は、系の特徴的な振動の周期と熱浴の振動の周期がおおよそ同等か、熱浴の方が長くなるように選ぶとよいとされています。熱浴の振動の周期と系の振動の周期の関係は、大雑把には次の式で評価できます (S. Nosé, Progress of Theoretical Physics Supplement No 103, 1991, pp.1-46)。

$$\tau = \frac{2\pi}{\omega} = 2\pi \left(\frac{Q}{2gk_{\text{B}}T} \right)^{1/2}$$

ここで τ , ω はそれぞれ系の周期と周波数, g は系の自由度($3 \times$ その熱浴に関連付けられている原子の数), k_{B} はボルツマン定数, T がその熱浴の温度です。例えば, τ を 0.05 ps, 原子の数を 8, 温度を 300 K として上式で Q の値を見積もると, 原子単位でおおよそ 4600 程度となります。PHASE/0 では, Q の値を直接指定することも、周期を介して間接的に指定することも可能となっています。質量の指定も周期の指定もない場合、周期が時間きざみ dt のおおよそ 50 倍となる Q の値が採用されます。

5.3 高精度な電子状態の計算

5.3.1 DFT+U 法

5.3.1.1 機能の概要

PHASE は、密度汎関数理論に基づき、ほとんどの物質の電子状態を高精度に計算することができますが、強相関電子系に対しては不正確な電子状態を与えることがあります。この原因として、密度汎関数法を適用する際に導入した局所密度近似の限界が知られています。この欠点を補う手段として、PHASE は LDA+U 法または DFT+U 法を実装しています。これは、局在電子間の斥力相互作用をオンサイトクーロン相互作用として取り込む手法です。

DFT+U 法にはいくつかの方法がありますが、PHASE では単純化された回転不変モデルを採用しています。このモデルでは、DFT+U のエネルギー汎関数 ($E_{\text{DFT+U}}$) は、局所密度近似のエネルギー汎関数 (E_{DFT}) と “+U” による補正エネルギー項の和として表されます。(後者の補正項は、ハバード補正項とも呼ばれます。) また、ハバード補正項は、各サイトにて計算された占有行列 ρ の関数です。

$$E_{\text{DFT+U}} = E_{\text{DFT}} + \frac{U_{\text{eff}}}{2} \sum_{l,m,\sigma} \left\{ \rho_{m,m}^{l\sigma} - \sum_{m'} \rho_{m,m'}^{l\sigma} \rho_{m',m}^{l\sigma} \right\}$$

ここで、 l 、 $m(m')$ 、及び σ は、原子サイト、磁気量子数、及びスピン指標に対応します。 U_{eff} は、有効的クーロン相互作用の大きさです。

占有行列は、局在軌道（原子軌道）に電子波動関数を射影することにより計算します。

$$\rho_{m,m'}^{l\sigma} = \sum_{k,n} f_{kn}^{\sigma} \langle \Psi_{kn}^{\sigma} | \phi_m^l \rangle \langle \phi_{m'}^l | \Psi_{kn}^{\sigma} \rangle$$

ここで、 k 及び n は、それぞれ波数ベクトル及びバンド指標です。また、 f_{kn}^{σ} は電子状態 $kn\sigma$ の占有数です。

ハバード補正は、局在した軌道の縮退したエネルギー準位の分裂をもたらします。特に、軌道が完全に占有（非占有）の場合、そのエネルギー準位は $\frac{U_{\text{eff}}}{2}$ だけ低下(上昇) します。なお、 U_{eff} の値は、実験に一致するように取るか、或いは文献値をもとに決めます。

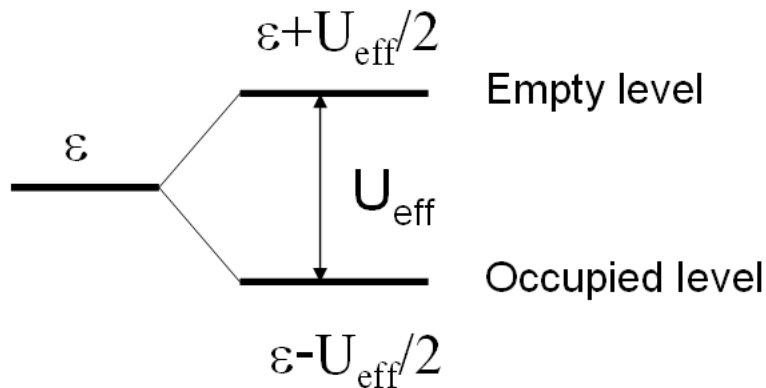


図 5-35 ハバード補正による軌道エネルギーの変化

5.3.1.2 入力パラメータ

DFT+U 法を使用するには、以下の手順が必要です。始めに、**accuracy** ブロック内に **hubbard** ブロックと **projector_list** ブロックを書き加えます。前者では有効クーロン相互作用エネルギーの値(**ueff**)を指定します。なお、**sw_hubbard=on** は、ハバード補正を行うことを宣言するために記述します。後者では、占有行列の計算で使用する原子軌道の有効半径を指定します。**no** はプロジェクター番号、**group** はプロジェクターのグループ番号、**radius** は有効原子半径、**l** は方位量子数です。**hubbard** ブロックで指定したプロジェクター番号は、**projector_list** ブロックで指定したプロジェクター番号に対応することにご留意ください。

```
accuracy{
  ...
  hubbard{
    sw_hubbard = on
    projectors{
      #units eV
      #tag no ueff
          1 10.0
    }
  }
  projector_list{
    projectors{
      #tag no group radius l
          1 1 2.75 2
    }
  }
  ...
}
```

次に、**structure** ブロックにて、ハバード補正を適用する原子を指定します。**proj_group** で指定する番号は、**accuracy** ブロックで定義したプロジェクター番号に対応しています。ハバード補正を行わない原子には、**proj_group** として0を割り当てます。なお、異種元素に対して、同一の **proj_group** 値を割り当てることは出来ません。

```
structure{
  ...
  atom_list{
    coordinate_system = internal ! {cartesian|internal}
    atoms{
      !#default mobile=no
      !#tag rx ry rz element proj_group
          0.0 0.0 0.0 Sr 0
          0.5 0.5 0.5 Ti 1
          0.0 0.5 0.5 O 0
          0.5 0.0 0.5 O 0
          0.5 0.5 0.0 O 0
    }
  }
  ...
}
```

DFT+U 法による計算は、しばしば局所極小へ電子状態が収束してしまいます。異なる波動関数ソルバーや電荷密度ミキサーによる計算を複数回行い、このような状況に陥っていないことを確認することを推奨します。

初期の占有行列の設定が不適切な場合、収束しづらかったり局所極小の電子状態へ至ってしまう場合があります。このようなことを避けるため、占有行列データファイル（後述の **occmat.data** ファイル）を介して初期占有行列

を手動で指定することも可能です。occmat.data ファイルを用意し、入力パラメータファイルに以下のような記述を行います。

```
accuracy{
  ...
  hubbard{
    initial_occmat = file
  }
  ...
}
```

occmat.data ファイルの書き方については、次節で説明します。

5.3.1.3 計算結果の出力

phase を実行します。

全エネルギーの出力とその成分の出力のあとにハバードエネルギー(HE)とハバードポテンシャルエネルギー(HP)が追加で出力されます。

TOTAL ENERGY FOR	2	-TH ITER=	-79.756461901287	edel =	0.482992D+01
KI=	45.2522902	HA=	125.6089055	XC=	-43.2979227
LO=	-147.0597534	EW=	-92.0686823	PC=	12.2272681
EN=	0.0000000	HP=	0.6709743		

また、ハバード補正を行った原子上の占有行列の各要素が出力されます。is はスピンの番号、ia は原子の番号、l は方位量子数を意味します。なお、占有行列の次元は $(2l+1) \times (2l+1)$ です。この行列の (m, m') 成分は、磁気量子数 m 及び m' ($1 \leq m, m' \leq 2l+1$) の原子軌道間における占有行列の要素に対応します。なお、各軌道の性格は、表 5.1 に表記しています。Printout ブロックで iprihubbard を 2 以上に設定している場合、占有行列を対角化することにより得られる、各原子軌道の占有数が出力されます。占有数は"'"の左側に、対応する固有ベクトルは右側に表示されます。

Occupation Mattrix: is, ia, l=						1	2	2
0.583	0.000	0.000	0.000	0.000				
0.000	0.583	0.000	0.000	0.000				
0.000	0.000	0.529	0.000	0.000				
0.000	0.000	0.000	0.529	0.000				
0.000	0.000	0.000	0.000	0.529				
Diagonalizing Occupation Mattrix: is, ia, l=						1	2	2
0.529:	0.000	0.000	0.000	-1.000	0.000			
0.529:	0.000	0.000	1.000	0.000	0.000			
0.529:	0.000	0.000	0.000	0.000	1.000			
0.583:	0.000	1.000	0.000	0.000	0.000			
0.583:	-1.000	0.000	0.000	0.000	0.000			

Occmat.data というファイルには、計算が終了する直前の SCF iteration における占有行列の要素が出力されます。

```
16 : num_om
.....
1   3   1   3   1 : is, ia, iproj; it, l
0.17441054E+01 -0.20464246E-02 -0.99899010E-03
-0.20464246E-02 0.17539484E+01 -0.39442624E-02
-0.99899010E-03 -0.39442624E-02 0.17529809E+01
1   4   1   3   1 : is, ia, iproj; it, l
```

0.17365161E+01	-0.12145064E-01	-0.11970673E-01
-0.12145064E-01	0.17903944E+01	-0.85524320E-02
-0.11970673E-01	-0.85524320E-02	0.17856965E+01
.....		

1 行目の **num_om** は、生成された占有行列の数を意味します。2 行目以降は、ハバード補正を行った原子における占有行列の要素が出力されます。**is** はスピンの番号、**ia** は原子の番号、**iproj** はプロジェクター番号、**it** は原子種の番号、**l** は方位量子数を意味します。出力される占有行列の数は、**num_om** に一致しています。

占有行列 $n^{\sigma,i,p}$ は $2l + 1$ 行 $2l + 1$ 列の行列形式で記述します。

$$\begin{array}{cccc}
n_{1,1}^{\sigma,i,p} & n_{1,2}^{\sigma,i,p} & \cdots & n_{1,2l+1}^{\sigma,i,p} \\
n_{2,1}^{\sigma,i,p} & n_{2,2}^{\sigma,i,p} & \cdots & n_{2,2l+1}^{\sigma,i,p} \\
\vdots & \vdots & \ddots & \vdots \\
n_{2l+1,1}^{\sigma,i,p} & n_{2l+1,2}^{\sigma,i,p} & \cdots & n_{2l+1,2l+1}^{\sigma,i,p}
\end{array}$$

行列の添え字は磁気量子数を表します。各方位量子数 l での添え字と軌道の性格との対応を表 5.1 に示します。**occmat.data** は計算を継続する場合や、占有行列の初期値を与える場合必要となります

表 5.1 占有行列の添え字と軌道の性格の対応表

占有行列の添え字	$l = 0$	$l = 1$	$l = 2$	$l = 3$
1	s	x	$3z^2 - r^2$	$z(5z^2 - 3r^2)$
2		y	$x^2 - y^2$	$x(5z^2 - 3r^2)$
3		z	xy	$y(5z^2 - 3r^2)$
4			yz	$z(x^2 - y^2)$
5			zx	xyz
6				$x(x^2 - 3y^2)$
7				$y(3x^2 - y^2)$

5.3.1.4 計算例：立方晶 SrTiO3

立方晶 SrTiO3 の計算例です。

- DFT+U/SrTiO3/cubic+u (U_{eff} は、Ti 3d 軌道に対して 10 eV)
- DFT+U/SrTiO3/cubic (U_{eff} は 0 eV)

これらの計算結果を図 5-36 に示します。

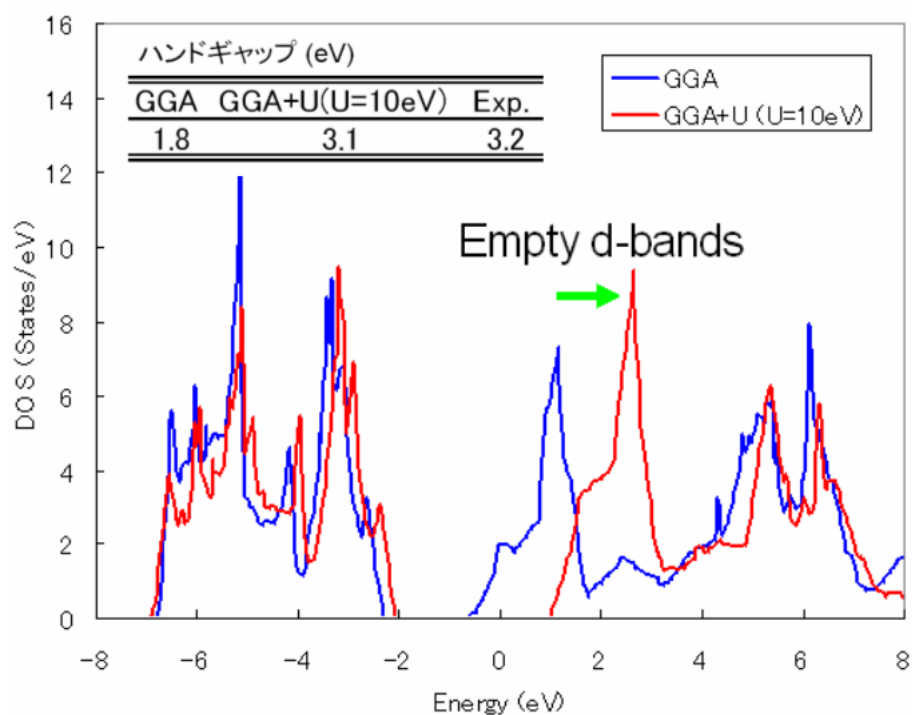


図 5-36 SrTiO3 の状態密度

5.3.1.5 計算例：立方晶 LaVO3

立方晶 LaVO3 の計算例です。

- DFT+U/LaVO3/cubic+u (U_{eff} は La 4f 軌道に対して 20 eV)
- DFT+U/LaVO3/cubic (U_{eff} は 0 eV)

U_{eff} が 0 eV のときは、4f 軌道によるバンドはフェルミレベルの上 1.5eV 上に現れますが、 U_{eff} を 20eV とすると 8eV 下に現れます。

5.3.1.6 計算例：斜方晶 LaVO3

斜方晶 LaVO3 の計算例です。

- DFT+U/LaVO3/orthrombic+u (U_{eff} は、V 3d 軌道は 5 eV、La 4f 軌道は 20 eV)
- DFT+U/LaVO3/orthrombic (U_{eff} は 0 eV)

DFT+U 計算では V 原子上の磁気モーメントが反強磁性的に配列します。

5.3.1.7 立方晶 FeO

立方晶 FeO の計算例です。この計算例は、occmat.data ファイル内の数字を占有行列の初期値として利用する

ものです。

- DFT+U/FeO/gga+u (U_{eff} は Fe 3d 軌道に対して 5 eV)
- DFT+U/FeO/gga (U_{eff} は 0 eV)

アップスピンに対しては占有行列の対角要素が 1、ダウンスピンに対しては $3z^2 - r^2$ 軌道以外の対角要素が 0 に設定されています。 U_{eff} が 0 eV では $3z^2 - r^2$ 軌道の性格を持つ d バンドがフェルミレベルより上に現れますが、ハバード補正では下に現れます。また、ハバード補正によりバンドギャップが開きます。

5.3.2 ハイブリッド汎関数

5.3.2.1 機能の概要

局所密度近似 (local density approximation=LDA) を改善する手法には一般勾配近似 (generalized gradientapproxiation=GGA) のほかに、Hartree-Fock 交換汎関数を一定量取り入れるハイブリッド汎関数法があります。PHASE/0 では、PBE0[1-4]と HSE06[5-8]の二種類の汎関数が使えます。

厳密交換エネルギー (Hartree-Fock 交換エネルギー) E_x^{exact} は、 $\{\mathbf{k}, \nu, \sigma\}$ で指定される {サンプリング \mathbf{k} 点、バンド、スピン状態} の波動関数 $\psi_{k\nu\sigma}(\mathbf{r})$ を用いて

$$E_x^{\text{exact}} = -\frac{1}{2} \sum_{\sigma} \sum_{k\nu, k'\nu'}^{\text{occ}} \int d\mathbf{r}_1 \int d\mathbf{r}_2 \left[\frac{\psi_{k\nu\sigma}^*(\mathbf{r}_1) \psi_{k'\nu'\sigma}(\mathbf{r}_1) \psi_{k'\nu'\sigma}^*(\mathbf{r}_2) \psi_{k\nu\sigma}(\mathbf{r}_2)}{|\mathbf{r}_1 - \mathbf{r}_2|} + \sum_{ij} \frac{Q_{ij}(\mathbf{r}_1 - \mathbf{r}_2) \langle \psi_{k\nu\sigma} | \beta_i \rangle \langle \beta_j | \psi_{k'\nu'\sigma} \rangle}{|\mathbf{r}_1 - \mathbf{r}_2|} \right] \quad (1)$$

で与えられます。ここで、 ν, ν' に関する和は占有状態に限られます。また、2 行目の Q_{ij} を含む項は欠損電荷に由来し、ウルトラソフト擬ポテンシャル利用時に現れます。

PBE0 汎関数は α をパラメータとして

$$E_{xc}^{\text{PBE0}} = \alpha E_x^{\text{exact}} + (1 - \alpha) E_x^{\text{PBE}} + E_c^{\text{PBE}} \quad (2)$$

と定義されます[1-4]。ここで、 E_x^{PBE} は PBE 交換汎関数で、 E_c^{PBE} は PBE 相関汎関数です。 $\alpha = \frac{1}{4}$ がよく使われ

る値です。ハイブリッド汎関数には、このほかに交換汎関数 E_x^{exact} をスクリーニングして加える HSE 汎関数 E_{xc}^{HSE}

があります。これは、

$$E_{xc}^{\text{HSE}} = \alpha E_x^{\text{exact,SR}}(\omega) + (1 - \alpha) E_x^{\text{PBE,SR}}(\omega) + E_x^{\text{PBE,LR}}(\omega) + E_c^{\text{PBE}} \quad (3)$$

の形式で表されます[5-8]。 ω は短距離相互作用が働く範囲を制御する調整パラメータで、 $\omega = 0$ で E_{xc}^{PBE0} に等し

く、 $\omega \rightarrow \infty$ で漸近的に E_{xc}^{PBE} に近づきます。HSE06 汎関数を利用すると、分子の場合はセルサイズ、結晶の場合

は \mathbf{k} 点分割数に対する収束性が、PBE0 汎関数を利用する場合に比較してよくなります。つまり同じ収束性を得るための計算負荷が相対的に小さくなります。計算精度を保ちながら計算負荷を節約できる値として、 ω には 0.1(Bohr⁻¹)程度の値がよく使われます。

5.3.2.2 入力パラメータ

(1) 基本的な設定

Hybrid 汎関数で電子状態計算を行うには、以下のように指定します。

```
accuracy{
  ksampling{
    method = gamma
    base_reduction_for_GAMMA = OFF
    base_symmetrization_for_GAMMA = OFF
```

```

    }
    xctype = ggapbe
    hybrid_functional{
        sw_hybrid_functional = ON
        functional_type = HSE06 ! {PBE0 | HSE06 | HF}
        alpha = 0.25
        omega = 0.106
    }
}

```

この例の場合は、HSE06 汎関数を指定したことになります。HSE06 のほか、PBE0、HF を指定することが可能です。例では、sampling k 点は Γ 点 (method=gamma) を指定していますが、ほかに格子状 sampling (method=mesh) あるいは Monkhorst-Pack によるスペシャル k 点 sampling (method=monk) を指定することもできます。パラメータ alpha は(2)、(3)式に現れる厳密交換相互作用の混合率 α で、デフォルトの値は 0.25 です。Omega は(3)式に現れる遮蔽パラメータ ω で、デフォルト値は 0.106 bohr⁻¹ です。

相関相互作用を取り除いて、Hartree-Fock 計算を行う場合は以下のように設定します。

```

accuracy{
    hybrid_functional{
        sw_hybrid_functional = ON
        functional_type = HF
    }
}

```

ただし、Hartree-Fock 計算の収束は PBE0 と比べても著しく遅くなります。

PBE 汎関数を使って収束させた波動関数と電荷密度がある場合、これを hybrid 汎関数計算の初期波動関数と初期電荷とすることができます。そのための入力例を次に示します。必須ではありませんが、ハイブリッド汎関数法は通常の GGA よりもはるかに多くの計算時間がかかるので、このように設定し少しでも収束回数を減らすことを推奨します。

```

accuracy{
    initial_wavefunctions = file
    initial_charge_density = file
}

```

このように計算を遂行する場合、計算に先だって PBE 汎関数計算の波動関数ファイル(zaj.data)と電荷密度ファイル(nfchg.data)を作業ディレクトリに必ずコピーします。

ハイブリッド汎関数法は、バンドおよび k 点の 2 重ループの処理があります。このうち、内側の k 点に関するループは計算精度にさほど影響がない場合があります。そこで、この内側の k 点ループを“間引く”ことによって計算量を減らす機能が PHASE には備わっています。この機能を利用するには、以下のように記述します。

```

accuracy{
    ksampling{
        method = mesh
        mesh{
            nx = 4
            ny = 4
            nz = 4
        }
    }
    hybrid_functional{
        reduction_factor{
            f1 = 2
            f2 = 2
            f3 = 2
        }
    }
}

```


変数 **f1, f2, f3** によって **nx, ny, nz** をそれぞれ何分の一にするかを指定します。この例では、それぞれの方向の内側ループのメッシュ数が 1/2 になります。

(2) 3 軸並列版において並列化軸を入れ替える方法

3 軸並列版でハイブリッド汎関数法を利用する場合、通常の計算と異なる並列化軸を採用することによって大幅な高速化が達成できます。この機能を利用する場合、以下のように設定します。

```
accuracy{
  hybrid_functional{
    sw_change_axis = on
  }
}
```

この設定は、次に説明する高速化の設定を 3 軸並列版で利用する場合有効にする必要があります。

(3) メッシュ調整による高速化

※ 以下の機能は 2 軸並列版と 3 軸並列版において **sw_change_axis = on** とした場合においてのみ利用可能です

ウルトラソフト擬ポテンシャルを利用する場合、ハイブリッド汎関数計算の演算時間のほとんどがウルトラソフト擬ポテンシャル特有の電荷密度構成（欠損電荷密度。(1)式中の Q_{ij} ）に由来する処理に費やされます。ハイブリッド汎関数計算においては、欠損電荷の高周波数成分（Gの大きい成分）を省くことによって、計算精度に大きな影響を及ぼさずに高速化することが可能です。PHASE/0 でこの機能を利用するには、変数 **charge_mesh** を利用します。

```
accuracy{
  hybrid_functional{
    charge_mesh = fine
  }
}
```

変数 **charge_mesh** には、**exact, fine, moderate, coarse** のいずれかを指定します。**exact** を指定すると欠損電荷のすべての波数成分を考慮します。**fine, moderate, coarse** の順に考慮する成分が少なくなります。**charge_mesh** のデフォルト値は **fine** です。

※ 以下の機能は 3 軸並列版において **sw_change_axis = on** とした場合においてのみ利用可能です
波動関数の **FFT** メッシュを削減することも可能です。程度にもよりますが、このメッシュもやはりある程度は削減しても計算精度に大きな影響を及ぼしません。この機能を利用するには、以下のような設定を施します。

```
accuracy{
  ...
  hybrid_functional{
    sw_change_axis = on
    cutoff_wf_for_exx = 9 rydberg
  }
}
```

accuracy ブロックの下に **hybrid_functional** ブロックに **cutoff_wf_for_exx** を定義し、そこに採用したいカットオフエネルギーを指定します。ここで指定できるのは通常のカットオフエネルギー以下の値であり、それを超える値を指定した場合通常のカットオフエネルギーで置き換わります。**cutoff_wf_for_exx** のデフォルト値は通常のカットオフエネルギーと同じ値です。波動関数の **FFT** は、特にノルム保存型の擬ポテンシャルを採用している

場合は計算時間のかなりの割合を占めることになりますので、この設定によって演算量を減らすことは有効です。参考のため、サンプルの Si 結晶の場合に `cutoff_wf_for_exx` を通常のカットオフエネルギーの半分の値にした場合の全エネルギーと計算時間を報告します（時間は、通常のカットオフエネルギーの場合を 1 としています）。

	<code>cutoff_wf_for_exx = cutoff_wf</code>	<code>cutoff_wf_for_exx = cutoff_wf/2</code>
全エネルギー (ha)	-7.8457557283	-7.8457499896
計算時間	1	0.34

（４）実空間法による高速化

※ 以下の機能は 2 軸並列版と 3 軸並列版において `sw_change_axis = on` とした場合においてのみ利用可能です

ウルトラソフト擬ポテンシャルを利用している場合、欠損電荷に由来する処理を実空間で行うことによって高速化することも可能です。それには、次のように設定します。

```
accuracy{
  hybrid_functional{
    sw_rspace = on
  }
}
```

通常、ハイブリッド汎関数法における欠損電荷の処理は $O(N^4)$ の時間がかかりますが、実空間法を利用することによってこの時間を $O(N^3)$ にすることができるので、ある程度大きな系では必須の指定となります。上述の `charge_mesh` パラメータは、逆空間の場合と同じ様に利用することが可能です。

`sw_rspace = on` の場合に、処理を行列－行列積の形に変形し、Level3 BLAS によって処理させることができます。この機能は、以下の要領で利用します。

```
accuracy{
  hybrid_functional{
    sw_rspace = on
    sw_rspace_dgemm = on
  }
}
```

この設定によってさらなる高速化が実現できる場合があります。

5.3.2.3 計算例

（１）水素分子

水素分子の PBE 計算、PBE0 計算、Hartree-Fock 計算の計算例題は、`samples/hybrid/H2` 以下のディレクトリ PBE、PBE0、HF です。`go_h2.sh` を実行すると、これらの計算を順番に実行します。これらの結果と Gaussian03 の結果との比較を図 5-37 に示します。

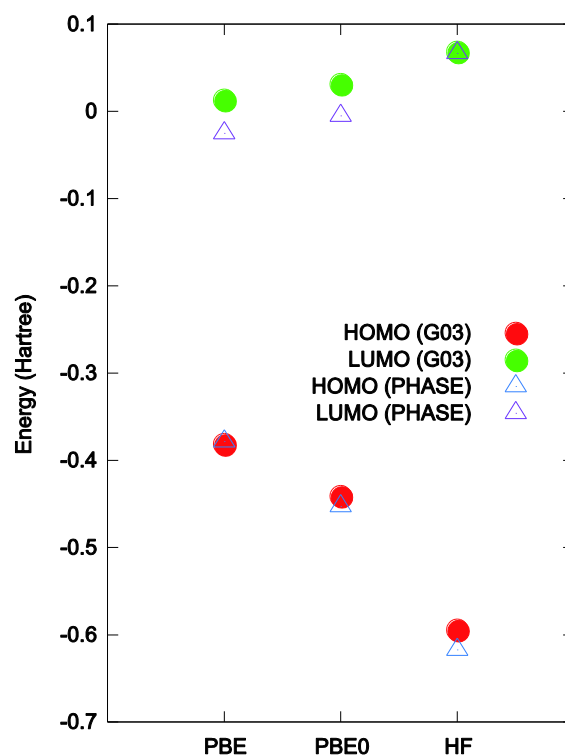


図 5-37 PBE 汎関数法, PBE0 汎関数法, Hartree-Fock 法による水素分子の HOMO 準位と LUMO 準位のエネルギーが Gaussian03 (G03) の結果と比較して示されている。

(2) 水分子

水分子の PBE 計算、PBE0 計算例題は、samples/hybrid/H2O 以下のディレクトリの PBE、PBE0 です。go_h2o.sh を実行すると、これらの計算を順番に実行します。これらの結果と Gaussian03 の結果との比較を図 5-38 に示します。

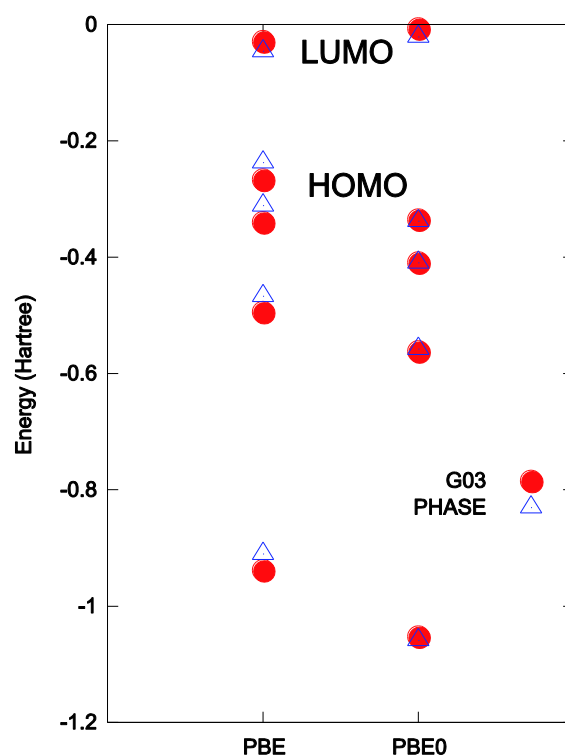


図 5-38 PBE 汎関数法, PBE0 汎関数法による水分子のエネルギー準位が Gaussian03 (G03) の結果と比較して示

されている。

(3) シリコン結晶

シリコン結晶の状態密度の計算を、PBE, PBE0, HSE06 で実行する例題が `samples/hybrid/Si` 以下にあります。計算を実行すると得られる状態密度図を図 5-39 に、得られるバンドギャップを表 5.2 に示します。図 5.37 に、得られるバンドギャップをに示します。図 5.35 に、得られるバンドギャップをに示します。図 5.37 に、得られるバンドギャップをに示します。図 5.35 に、得られるバンドギャップをに示します。図 5.37 に、得られるバンドギャップをに示します。図 5.35 に、得られるバンドギャップをに示します。図 5.37 に、得られるバンドギャップをに示します。

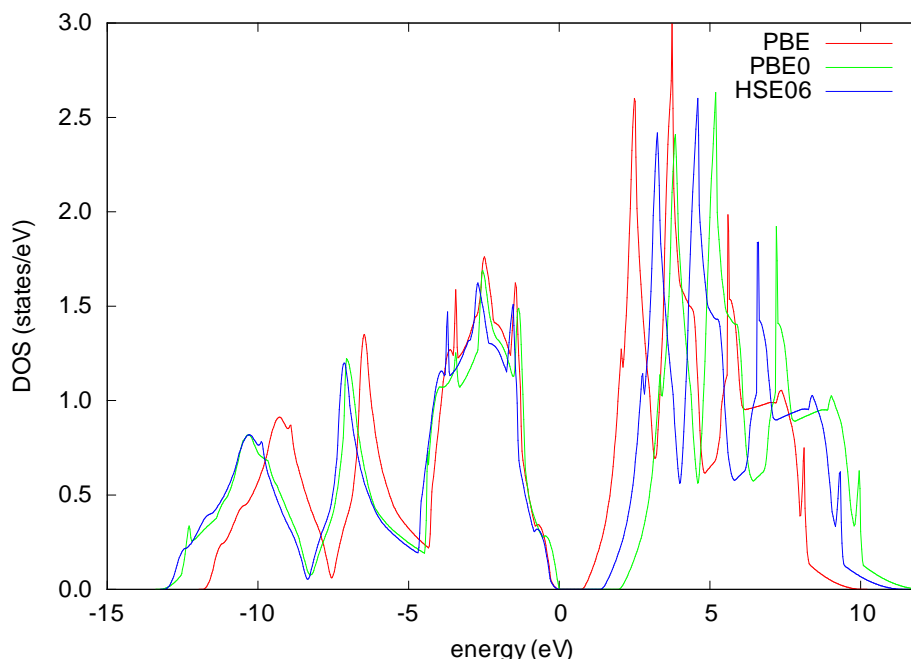


図 5-39 PBE, PBE0, HSE06 汎関数によって得られた Si 結晶の状態密度

表 5.2 各汎関数によって得られるバンドギャップの比較

汎関数	バンドギャップ (eV)
PBE	0.7
PBE0	1.9
HSE06	1.3

`samples/hybrid/Si_k10` は、k 点メッシュを $10 \times 10 \times 10$ とした例題です。非常に時間のかかる計算ですが、より精密な状態密度図が得られます。

5.3.2.4 使用における注意点

- ハイブリッド汎関数法は反転対称性を利用する高速化機能が使えません。反転対称性を有する系であっても `sw_inversion` パラメータを `on` とすると計算が破綻しますので、この設定はしないでください。

5.3.2.5 参考文献

- [1] J. P. Perdew, M. Emzerhof, and K. Burke, J. Chem. Phys. **105** (1996) 9982.
- [2] M. Emzerhof, J. P. Perdew, and K. Burke, Int J. Quantum Chem. **64** (1997) 285.
- [3] M. Emzerhof and G. E. Scuseria, J. Chem. Phys. **110** (1999) 5029.
- [4] C. Adamo and V. Barone, J. Chem. Phys. **110** (1999) 6158.

- [5] J. Heyd, G. E. Scuseria, and M. Ernzerhof, J. Chem. Phys. **118** (2003) 8207.
- [6] J. Heyd and G. E. Scuseria, J. Chem. Phys. **120** (2004) 7274.
- [7] J. Heyd and G. E. Scuseria, J. Chem. Phys. **121** (2004) 1187.
- [8] J. Heyd, G. E. Scuseria and M. Ernzerhof, J. Chem. Phys. **124** (2006) 219906.

5.3.3 SC-DFT 法

5.3.3.1 機能の概要

ハイブリッド汎関数法とは密度汎関数理論の交換相関相互作用に厳密交換相互作用を混合する計算手法ですが、その混合の仕方については様々な方法が提案されています。最も素朴な手法は PBE0 と呼ばれる手法です。PBE0 法では、一定の値 α で厳密交換相互作用を混合します。通常適用される α の値は 0.25 ですが、これを系に応じた最適な割合を第一原理的に導く手法が SC-DFT 法です (Jonathan H. Skone, 1 Marco Govoni, and Giulia Galli, “Self-consistent hybrid functional for condensed systems”, Phys. Rev. B **89** (2014) 195112). SC-DFT 法は、誘電率と混合比を結び付け、得られる誘電率と混合比が自己無撞着になるまで繰り返し計算を行うという計算手法になっています。

5.3.3.2 理論

ハイブリッド汎関数法においては、最も単純な PBE0 法の場合以下のように交換相互作用を計算します。

$$E_{\text{hybrid}}^x = (1 - \alpha) E_{\text{PBE}}^x + \alpha E_{\text{exact}}^x. \quad (5-2)$$

ここで E_{PBE}^x は GGA-PBE の交換相互作用、 E_{exact}^x は厳密交換相互作用です。(5-2)式のパラメーター α の妥当な値は自明ではなく、また系によって異なると考えられます。このパラメーターを、以下のように近似するのが SC-DFT 法です。

$$\alpha = \frac{1}{\epsilon} \quad (5-3)$$

ϵ は誘電率です。SC-DFT 法では、自己無撞着な α と ϵ をもとめるために繰り返し計算を行います。すなわち、初期 α を設定し、誘電率を計算します。得られた誘電率から(5-3)式を利用して α を求め、最初の α と同じ結果ならば収束したとみなし得られた α を最適なものとします。収束していない場合は得られた α を使って誘電率計算を再実行します。このような手続きを、 α が収束するまで繰り返します。

5.3.3.3 使い方

SC-DFT 法を利用するためには、以下のように control ブロックにおいて変数 driver の値を sc_dft とします。

```
control{
  driver = sc_dft
}
```

SC-DFT の収束判定条件は、以下のように設定します。

```
accuracy{
  sc_dft{
    delta_epsilon = 0.01
  }
}
```

delta_epsilon のデフォルト値は 0.01 です。

SC-DFT 計算は誘電率計算を行います。デフォルトの設定で問題ないのであればそのための設定は必要ありません。デフォルトの設定を変更する場合は epsilon ブロックを挿入し、そこで設定を行います。ただし、SC-DFT 法の場合 photon ブロックにおいて polar, Poynting ベクトルの値はゼロにする必要があります。すなわち、以下のように設定する必要があります。

```
epsilon{
  ...
  photon{
```

```

    poynting{
        ux = 0.0, uy = 0.0, uz = 0.0
    }
    polar{
        px = 0.0, py = 0.0, pz = 0.0
    }
}
}

```

誘電関数計算は、四面体法によって行われます。そのため、以下のように **k** 点サンプリングを四面体法が利用できるように設定する必要があります。

```

accuracy{
    ksampling{
        method = mesh
        ...
    }
    smearing{
        method = tetrahedral
    }
}

```

このように設定しないと不正な誘電率が得られてしまいます。

また、ハイブリッド汎関数法による計算を行います。これは前節で説明したように設定してください。ただし、**functional_type** には何も指定しないか、**PBE0** を指定するようにしてください。

計算の実行は、通常通り行います。通常の計算と違い収束する度に新しい計算が行われます。混合比はログファイルに記録されます。以下の要領でその値を調べることができます。

```

$ grep alpha_exx output000
!!** alpha_exx = 0.1425
!!** alpha_exx = 0.1208
!!** alpha_exx = 0.1113
!!** alpha_exx = 0.1081
!!** alpha_exx = 0.1080

```

また、**nfefn.data** ファイルには以下のように各 **SC-DFT** ステップにおけるエネルギーの履歴が記録されます。

iter_scdft	iter_ion	iter_total	etotal	forcmx
1	1	9	-7.8461877466	0.0000000000
2	1	12	-7.8443931212	0.0000000000
3	1	15	-7.8437399027	0.0000000000
4	1	18	-7.8434571365	0.0000000000
5	1	21	-7.8433616238	0.0000000000

ハイブリッド汎関数法は計算時間が膨大になりえるので、継続計算をする必要がでてくる場合もあるでしょう。継続計算を行うにあたって、特別なことを考慮する必要はありません。通常通り **control** ブロックの **condition** を **continuation** もしくは **automatic** とした上で **PHASE/0** を再実行してください。

5.3.3.4 計算例

Si 結晶に **SC-DFT** 法を適用した例を紹介します。Si 結晶は、通常の **PBE** 計算の場合バンドギャップが **0.6 eV**

程度と過小評価されることが知られています。また、PBE0 ($\alpha=0.25$)を用いると 1.94 eV 程度と過大評価されます。

まず、図 5-40 に混合比 α と誘電率の履歴を報告します。図から明らかなように、SC-DFT iteration 4 回目ではやくも収束しています。Si 結晶に限らず、SC-DFT 法の収束は速い場合が多いようです。得られた α の値は 0.108, 対応する誘電率は 9.26 です。

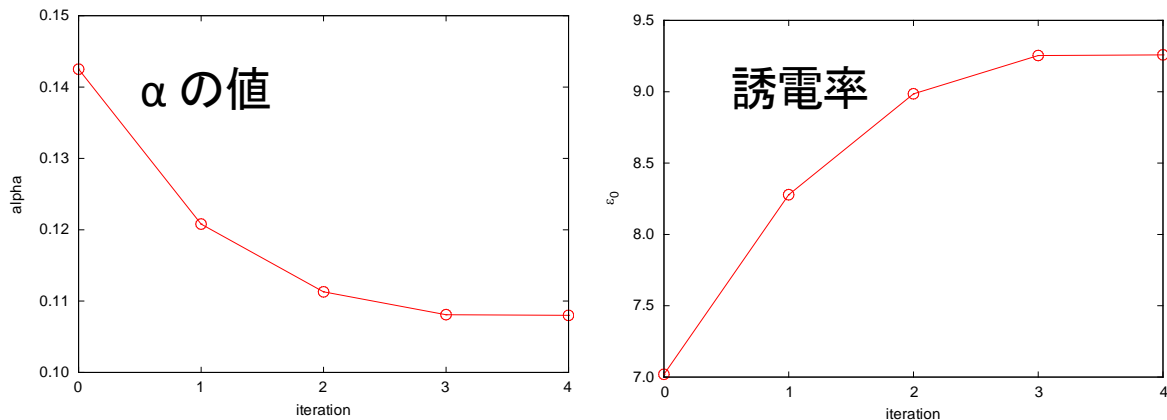


図 5-40 Si 結晶に対して SC-DFT 法を適用した例。左図： α の値の履歴，右図：誘電率の履歴。

つぎに、得られた α を利用して、状態密度の計算を行ってみました。結果を図 5-41 に報告します。比較のため PBE で求めた場合と $\alpha=0.25$ で求めた場合の結果も同時にプロットしました。

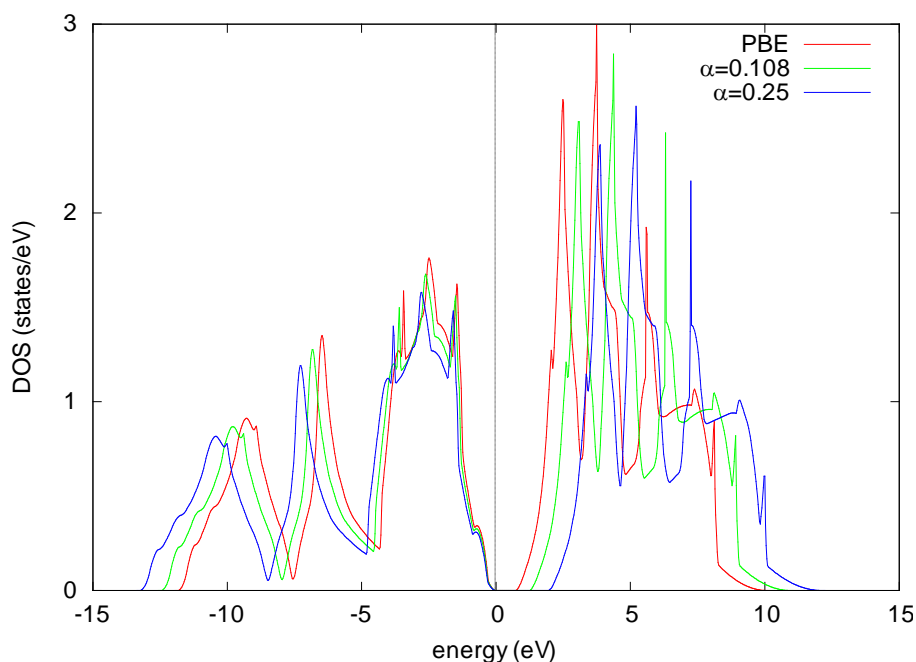


図 5-41 Si 結晶の状態密度

この状態密度より得られるバンドギャップは 1.23 eV です。これは、実測値の 1.17 eV に極めて近い値であるといえます。

5.3.4 ファンデルワールス相互作用（非局所相関項）

5.3.4.1 機能の概要

非局所相関項を第一原理的に計算する手法（van der Waals density functional (vdW-DF)）を利用した計算機能について説明します。PHASE でも採用されている一般化された密度勾配近似：Generalized Gradient Approximation (GGA)では非局所相関項が考慮されていないために、例えば積層グラファイトの層間凝集エネルギーなどを正確に計算することができません。本節で取り上げる計算機能は GGA のこの欠点を補うために用意されたもので、これを用いることで van der Waals 相互作用が大きく寄与する系の全エネルギーや電子状態もより正確に計算できるようになります。また、この vdWDF は第一原理的な手法を利用して経験的なパラメータ等を用いていないので、任意の形状の系に対して簡単に適用することができます。

5.3.4.2 非局所相関項を含めた全エネルギー計算

(1) 理論概要

● 基本理論

プログラム vdW.F90 では非局所相関項 E_c^{nl} を計算します。この E_c^{nl} に、GGA で得られる交換項 E_x^{GGA} と LDA から得られる交換項 E_c^{LDA} を足し合わせることで「非局所相関項も考慮した交換相関項」を導出します。すなわち、交換相関エネルギー E_{xc} は

$$E_{xc} = E_x^{GGA} + E_c^{LDA} + E_c^{nl} \quad (1)$$

となります。このうち右辺第3項の計算が最も困難で、vdW では Dion ら[1]によって開発された理論手法に習って数値計算されます。この理論手法では非局所相関項を

$$E_c^{nl} = \frac{1}{2} \int d\mathbf{r}_i d\mathbf{r}_k \rho(\mathbf{r}_i) \phi(\mathbf{r}_i, \mathbf{r}_k) \rho(\mathbf{r}_k) \quad (2)$$

として計算します。被積分関数に位置変数が2つ (\mathbf{r}_i と \mathbf{r}_k) あるこの式では、GGA や LDA と違って、離れた位置にある電荷密度同士 ($\rho(\mathbf{r}_i)$ と $\rho(\mathbf{r}_k)$) の相互作用も考慮しています。2変数関数 $\phi(\mathbf{r}_i, \mathbf{r}_k)$ は

$$\phi(\mathbf{r}_i, \mathbf{r}_k) = \frac{2}{\pi^2} \int_0^\infty da db a^2 b^2 W T \quad (3)$$

のように書けます。ここで、

$$W(a, b) = \frac{2}{a^3 b^3} [(3 - a^2)b \cos b \sin a + (3 - b^2)a \cos a \sin b + (a^2 - b^2 - 3) \sin a \sin b - 3ab \cos a \cos b] \quad (4)$$

です。また、

$$T[x_i(a), x_i(b), x_k(a), x_k(b)] = \frac{1}{2} \left[\frac{1}{x_i(a) + x_i(b)} + \frac{1}{x_k(a) + x_k(b)} \right] \times \left[\frac{1}{(x_i(a) + x_k(a))(x_i(b) + x_k(b))} + \frac{1}{(x_i(a) + x_k(b))(x_i(b) + x_k(a))} \right] \quad (5)$$

と定義されます。さらに各変数は

$$x_j(a) = \frac{a^2}{2} \times \frac{1}{1 - \exp\left(-\frac{4\pi a^2}{9d_j^2}\right)}, \quad (6)$$

$$d_j = |\mathbf{r}_i - \mathbf{r}_k| q_0(\mathbf{r}_j), \quad (7)$$

$$q_0(\mathbf{r}_j) = -\frac{4\pi}{3} \epsilon_{xc}^{LDA} \rho(\mathbf{r}_j) - \frac{Z_{ab}}{9} \left\{ \frac{\nabla \rho(\mathbf{r}_j)}{2k_F(\mathbf{r}_j)\rho(\mathbf{r}_j)} \right\}^2 k_F(\mathbf{r}_j), \quad (8)$$

$$k_F^3(\mathbf{r}_j) = 3\pi^2 \rho(\mathbf{r}_j) \quad (j = i \text{ or } k) \quad (9)$$

となっており、これからわかるように電荷密度分布を唯一の入力情報とした汎関数となるように設計されています。ここで式(8)の定数 $Z_{ab} = -0.8491$ は第一原理的に決定された係数で、vdwdf version 1 で採用されています。vdwdf version 2 では、この値を 2.2 倍した値が用いられます。局所密度近似による交換相関エネルギー密度 ϵ_{xc}^{LDA} は O. Gunnarsson *et al* [2]によるものを用いています。これら一連の式は plasmon-pole model をもとに設計されたものであり、そのため van der Waals 相互作用に代表される非局所相関項を含む全エネルギーが比較的低計算コストで非経験的に得られるようになっています。

さらに効率的に数値計算を行うために、式(2)の2変数関数 $\phi(\mathbf{r}_i, \mathbf{r}_k)$ を計算するアルゴリズムが変更されました。

この2変数関数 $\phi(\mathbf{r}_i, \mathbf{r}_k)$ は直接には d_i と d_j にしか依存しないため、 $d_i = D(1+\delta)$, $d_j = D(1-\delta)$ と新たに定義された2変数 D , δ を用いて $\phi(\mathbf{r}_i, \mathbf{r}_k)$ を $\phi(D, \delta)$ として予め計算された数値セットを用意するようにしました。これによって用意したグリッド点数に応じて式(3)の2重積分を逐一行う必要がなくなるため数値計算量が大幅に削減されています。

● 特異点周辺の数値積分

式(6)で変数 $a = 0$ でかつ $\mathbf{r}_i = \mathbf{r}_k$ の場合は数値計算によって $x_j(a)$ を決定するのは困難です。このため式(2)の数値積分も特異点を含むことになるため、難しくなります。そこで $|\mathbf{r}_i - \mathbf{r}_k| \ll 1$ の領域では電荷密度を $\rho(\mathbf{r}_i) = \rho(\mathbf{r}_k)$ と仮定して \mathbf{r}_k 積分の外に出すことにします。これによって式(2)のうち、 \mathbf{r}_i を中心にした微小半径 η_i の球内での \mathbf{r}_k 積分を

$$\begin{aligned} \frac{1}{2} \int_{\eta_i} d\mathbf{r}_k \rho(\mathbf{r}_i) \phi(\mathbf{r}_i, \mathbf{r}_k) \rho(\mathbf{r}_k) &\cong \frac{1}{2} \frac{4\pi\eta_i^3}{3} \rho^2(\mathbf{r}_i) \int_{\eta_i} d\mathbf{r}_k \phi(\mathbf{r}_i, \mathbf{r}_k) \rho(\mathbf{r}_k) \\ &= \frac{1}{2} \frac{4\pi\eta_i^3}{3} \rho^2(\mathbf{r}_i) \times 4\pi \int_0^{\eta_i} dr_{ik} \phi(d_i, d_k) \\ &= \frac{1}{2} \frac{4\pi\eta_i^3}{3} \rho^2(\mathbf{r}_i) \times 4\pi \int_0^{\eta_i q_0} dD \phi(D) \frac{D^2}{q_0^3} \end{aligned} \quad (10)$$

と単純にすることができます。ただし、 $r_{ik} = |\mathbf{r}_i - \mathbf{r}_k|$ としています。ここで2行目から3行目へは、 $D \equiv q_0 r_{ik} (= d_i = d_k)$ を定義して変数変換を用いています。3行目の被積分関数は特異点を含まないため、これで数値積分を実行することができます。

● 無限周期系への拡張

本手法では電荷密度分布情報を実空間表記で入力し、そのまま計算を進めていくためにこのままでは有限サイズの計算対象に向けた手法となっています。逆に無限周期系を計算するには十分に \mathbf{r}_i と \mathbf{r}_k が離れた場合まで式(2)の数値積分を行わなくてはならず、非常に計算効率が悪くなります。特に van der Waals 相互作用は遠距離においてもその寄与はなだらかにしか減衰しません。そこで漸近関数を用いることで、本手法をこの無限周期系にも対応できるようにしました。式(2)の被積分関数にある2変数関数 $\phi(\mathbf{r}_i, \mathbf{r}_k)$ は \mathbf{r}_i と \mathbf{r}_k が十分に離れた場合には

$$\begin{aligned} \phi(\mathbf{r}_i, \mathbf{r}_k) &\rightarrow -\frac{12 \left(\frac{4\pi}{9}\right)^3}{d_i^2 d_k^2 (d_i^2 + d_k^2)} \\ &= -\frac{C}{r_{ik}^6} \times \frac{1}{q_0^2(r_i) q_0^2(r_k) (q_0^2(r_i) + q_0^2(r_k))} \\ &= -\frac{C}{r_{ik}^6} \times \psi[\rho(\mathbf{r}_i) \rho(\mathbf{r}_k)] \end{aligned} \quad (11)$$

のような漸近関数に近似できることがその定義からわかります。ただし $C = 12(4\pi/9)^3$ 。この漸近関数には式(3)のような積分演算が含まれていないため非常に簡単に数値計算が行えるという利点があります。この式の3行目のうち \times の前の分数は r_k の増加に応じて単純に減衰していく漸近項で、これに対して \times の後ろの $\psi[\rho(\mathbf{r}_i), \rho(\mathbf{r}_k)]$ は式(8)からわかるように直接的には電荷密度しか変数を持たないため、周期系物質においては周期的な項となっています。このことを考慮すると、式(2)のうち、 \mathbf{r}_i と \mathbf{r}_k が十分に離れた場合($r_{ik} > \eta$)には次のように積分を単純化できます。すなわち

$$\begin{aligned} \frac{1}{2} \int_{r_{ik} > \eta} d\mathbf{r}_i d\mathbf{r}_k \rho(\mathbf{r}_i) \phi(\mathbf{r}_i, \mathbf{r}_k) \rho(\mathbf{r}_k) &\cong \frac{1}{2} dv^2 \sum_{\substack{\mathbf{r}_i \in \\ \text{unitcell}}} \sum_{\substack{\mathbf{r}_k \in \\ (r_{ik} > \eta)}} \rho(\mathbf{r}_i) \phi(\mathbf{r}_i, \mathbf{r}_k) \rho(\mathbf{r}_k) \\ &= \frac{1}{2} dv^2 \sum_{\substack{\mathbf{r}_i, \mathbf{r}_k \in \\ \text{unitcell}}} \sum_{\substack{\mathbf{t}_{xyz} \in \\ (|\mathbf{r}_k + \mathbf{t}_{xyz} - \mathbf{r}_i| > \eta)}} \rho(\mathbf{r}_i) \phi(\mathbf{r}_i, \mathbf{r}_k + \mathbf{t}_{xyz}) \rho(\mathbf{r}_k + \mathbf{t}_{xyz}) \\ &= -\frac{C}{2} dv^2 \sum_{\substack{\mathbf{r}_i, \mathbf{r}_k \in \\ \text{unitcell}}} \rho(\mathbf{r}_i) \psi[\rho(\mathbf{r}_i), \rho(\mathbf{r}_k)] \rho(\mathbf{r}_k) \sum_{\substack{\mathbf{t}_{xyz} \in \\ |\mathbf{r}_k + \mathbf{t}_{xyz} - \mathbf{r}_i| > \eta}} \frac{1}{r_{xyz}^6} \end{aligned} \quad (12)$$

と関数を分けることができます。ここで dv は数値計算を行う際のグリッド点に囲まれる最小直方体体積で、 \mathbf{t}_{xyz} は格子点同士を結ぶ結晶ベクトルです。この式3行目の総和のうち左側は \mathbf{r}_i と \mathbf{r}_k の範囲がユニットセル内に限定されているので、計算コストは限定的です。一方、右側は $1/r_{xyz}^6$ が実質無視できるほど小さくなるまで広範囲に

渡って総和することになります。しかし、そもそも関数型が非常に単純な上に、電荷密度分布 $\rho(\mathbf{r})$ とは無関係で、グリッド点の空間配置のみによって決まる値なので予め求めておけばよいものです。

以上のように漸近関数を使ってさらに式(12)のように積分順序を工夫することによって、実質無限遠の寄与まで効率よく考慮することができ、これによって本手法を無限周期系にも適用できるようにしています。

● 3 次スプライン補間と畳み込み積分を利用した高速化

Dion らによるももとの手法では、空間の 2 重積分を直接行うため膨大な演算量が必要とされます。そこで、Román-Pérez と Soler によって、スプライン補間と畳み込み積分を利用したより高速なアルゴリズムが開発されました(Guillermo Román-Pérez and José M. Soler, Phys. Rev. Lett. **103** 096102 (2009))。PHASE/0 はこの方法によって実際の計算を行います。

この方法では、まずカーネル関数を 3 次のスプライン補間によって展開します。

$$\begin{aligned}\phi(d, d') &= \phi(q|\mathbf{r}-\mathbf{r}'|, q'|\mathbf{r}-\mathbf{r}'|) \\ &= \sum_{\alpha\beta} \phi(q_\alpha|\mathbf{r}-\mathbf{r}'|, q_\beta|\mathbf{r}-\mathbf{r}'|) p_\alpha(q) p_\beta(q').\end{aligned}$$

$\theta_\alpha(\mathbf{r}) \equiv p_\alpha(q)\rho(\mathbf{r})$ を導入すると、2 重積分は以下のように畳み込み積分に変形することができます。

$$\begin{aligned}E_c^{\text{nl}} &= \frac{1}{2} \sum_{\alpha\beta} \iint d\mathbf{r} d\mathbf{r}' \theta_\alpha(\mathbf{r}) \theta_\beta(\mathbf{r}') \phi_{\alpha\beta}(|\mathbf{r}-\mathbf{r}'|) \\ &= \frac{\Omega}{2} \sum_\alpha \sum_{\mathbf{G}} \theta_\alpha^*(\mathbf{G}) \theta_\beta(\mathbf{G}) \phi_{\alpha\beta}(|\mathbf{G}|), \\ \theta_\alpha(\mathbf{G}) &= \int d\mathbf{r} \theta_\alpha(\mathbf{r}) \exp[-i\mathbf{G} \cdot \mathbf{r}], \\ \phi_{\alpha\beta}(|\mathbf{G}|) &= \frac{4\pi}{|\mathbf{G}|} \int r \phi_{\alpha\beta}(r) \sin[|\mathbf{G}|r] dr.\end{aligned}$$

このように変形すると、上式から分かるように高速フーリエ変換を利用して処理できるようになり、大幅な高速化が達成できます。

SCF 計算に組み込む場合、通常通りエネルギーを電子密度で汎関数微分します。

$$\begin{aligned}v_c^{\text{nl}}(\mathbf{r}) &= \sum_\alpha \left(u_\alpha(\mathbf{r}) \frac{\delta \theta_\alpha(\mathbf{r})}{\delta \rho(\mathbf{r})} + \sum_{\mathbf{r}'} u_\alpha(\mathbf{r}') \frac{\delta \theta_\alpha(\mathbf{r}')}{\delta |\nabla \rho(\mathbf{r}')|} \frac{\delta |\nabla \rho(\mathbf{r}')|}{\delta \rho(\mathbf{r})} \right), \\ u_\alpha(\mathbf{r}) &= \sum_\beta \sum_{\mathbf{r}'} \theta_\beta^*(\mathbf{r}') \phi_{\alpha\beta}(|\mathbf{r}-\mathbf{r}'|).\end{aligned}$$

$u_\alpha(\mathbf{r})$ も畳み込み積分になっているので、逆空間で計算し、逆 FFT を行うことによって計算することが可能です。

● アルゴリズム

非局所相関項 (van der Waals 項) を含めた全エネルギー計算は、交換相互作用は GGA、局所的な相関相互作用は LDA、そして非局所的な相関相互作用はここまで説明した vdWDF 理論を利用して行われます。計算方法としては、この計算をすべてセルフコンシステントに行う方法と、交換相互作用は GGA、相関相互作用は LDA を採用した変則的な汎関数を利用して収束解を得たあとにポスト处理的に vdWDF 効果を取り込む、という方法があります。PHASE/0 では前者を“SCF 版”、後者を“ワンショット版”と呼んでいます。ワンショット版は最後に 1 度 vdW の処理を行うのみなので、SCF 版と比較すると高速です。エネルギーを求めることのみが目的の場合ワンショット版でも多くの場合十分な精度が得られます。ただし、vdW エネルギー由来の原子間力も計算したい場合は SCF 版を採用する必要があります。

なお、上記の交換相互作用として、いくつかの GGA 汎関数が考えられています。さらに、これと Z_{ab} の値 (vdwdf のバージョン) の組み合わせにより、複数の vdWDF 汎関数が提案されています。PHASE/0 で使用可能な vdWDF の一覧を以下に示します。

名称	vdwdf のバージョン	交換相互作用
vdwdf	1	revpbe
vdwdf2	2	pw86r
vdwdf-c09x	1	c09x
vdwdf2-c09x	2	c09x
vdwdf-optpbe	1	optpbe
vdwdf2-b86r	2	b86r
vdwdf-cx	1	lvpw86r

(2) 使用方法

利用の前提として、GGAPBE 汎関数に対応した擬ポテンシャルファイルを用意する必要があります (公開擬ポテンシャルは、ほぼすべて GGAPBE に対応しています)。その上で、以下のような設定を行います。

```
accuracy{
  xctype = vdwdf2          ! vdwdf, vdwdf2, ... ,vdwdf-cx が選択可能。
  vdwdf{
    mode = scf
  }
}
```

accuracy ブロックの下に xctype 変数に、使用する vdWDF 汎関数名を指定します。さらに、vdwdf ブロックの下の変数 mode に SCF 版を利用する場合は scf を、ワンショット版を利用する場合は oneshot を指定します。変数 mode のデフォルト値は oneshot です。ただし、xctype = vdwdf 以外はワンショット計算には非対応です。

(3) 出力

結果は、通常の計算と変わりません。nfeffn.data ファイルにはエネルギーの履歴が、nfdynm.data ファイルには原子座標の履歴が記録されます。ワンショット版を利用している場合は、output000 ファイルに記録される途中のエネルギーと nfeffn.data ファイルで報告されるエネルギーが異なる点には注意が必要です。ワンショット版の場合、途中に記録されるエネルギーは交換相互作用に GGA、相関相互作用に LDA を利用した変則的な汎関数による結果だからです。ワンショット版の場合、計算収束し、vdWDF の計算が終わったあと、以下のように結果が output000 ファイルに報告されます。

```
vdW energy      :      0.0668443126 hartree
--> total energy :     -22.8808219651 hartree
```

nfeffn.data ファイルに記録されるエネルギー値は、vdWDF のエネルギーも含んだエネルギー値です。

5.3.4.3 計算例 1 : 積層グラファイトの全エネルギー計算

(1) はじめに

ここでは実際に本ルーチン (vdW) を用いた vdW-DF 計算の例を挙げます。計算対象は GGA や LDA では正確に計算できない典型例である、積層グラファイト (A-B stacking) の全エネルギーの層間距離依存性としまし

た．通常の GGA の範囲内でこの系の全エネルギー曲線を計算するとエネルギー的に安定な平衡点は現れず，結果的に各グラファイト層は互いに無限遠まで離散するという解釈になってしまいます．本来は適当な層間距離でエネルギー的に明確な安定点が存在するため，GGA によるこの解釈は定性的に間違っています．これは，非局所的な相互作用である van der Waals 相互作用を GGA が全く考慮できていないことに主な要因があり，本計算機能を利用することでこの間違いが修正されることを確認します．

(2) 計算条件

本計算では，実験値や他の理論計算の報告例が豊富にある A-B stacking 型の積層グラファイトを対象にしました．ユニットセル内に 8 個（2 層分）の炭素原子を含みます．グラファイトの積層方向に z 軸をとってセルサイズは $4.3 \times 2.5 \times d (=x \times y \times z [\text{\AA}^3])$ とし， z を 5 から 12[Å]まで変化させながら各 z 値での全エネルギーを計算させました．この計算を通常の GGA と vdW-DF 計算（`xctype=vdwdf, mode=oneshot`）の 2 通りで行い，それぞれから得られる全エネルギー曲線を比較します．vdW-DF 計算には入力情報として電荷密度分布 $\rho^{\text{GGA}x}(\mathbf{r})$ が必要になりますが，このグリッド密度は GGA 計算時に設定した `cutoff` 値に依存しており，ここでは $32 \times 18 \times 40$ (to 96) 個のグリッドを採用しています．なお，グリッド点数は整数であるため， z 軸の変化に伴って不連続に変化することになります．つまり，結果のエネルギー曲線が不自然に変化する場合はこの不連続性が原因であるため，必要に応じて PHASE の `inputfile` の `cutoff` 値を上げることでなめらかになります．また，vdW は非局所的な計算になるため GGA よりは大きな計算コストが必要となります．

(3) 計算結果

下図は上記条件で計算した積層グラファイト（A-B stacking）の全エネルギーの層間距離依存性を示したものです．赤線は通常の GGA による結果を，緑線は vdW による vdW-DF 計算の結果をそれぞれ示しています．また，青点は実験[3,4]による平衡点とそのときの凝集エネルギー（青線は誤差）を，同様に紫点は別の理論計算[5]によるものです．GGA 計算では全く平衡点が確認できないのに対して，vdW-DF 計算ではかなり実験値の近くで極小点を迎えているのが確認できます．

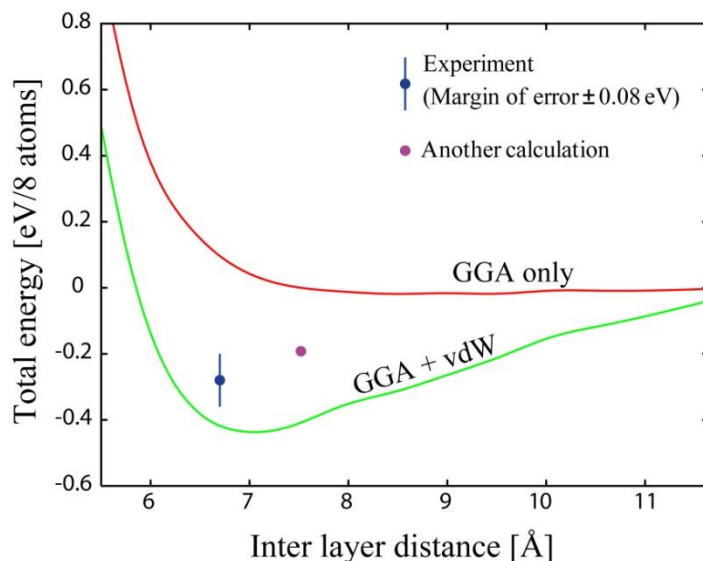


図 5-42 積層グラファイトの全エネルギーの層間距離依存性. GGA（赤線）と GGA+vdW（緑線）による比較

5.3.4.4 計算例 2 : 積層 2H-MoS₂ の全エネルギー計算

層状物質である 2H-MoS₂ (単位胞 6 原子) の全エネルギー計算を、*c* 軸方向の格子長をパラメーターにして行います。*a*、*b* 軸の格子長は 3.1612 Å で固定します。また、mode=scf により SCF 計算を行い、構造最適化を行います。結果は、以下に示すように、xctype = vdwdf よりも vdwdf2-c09x の方が、実験の格子長 12.2985 Å に近い *c* 軸長でエネルギーが最小となっています。

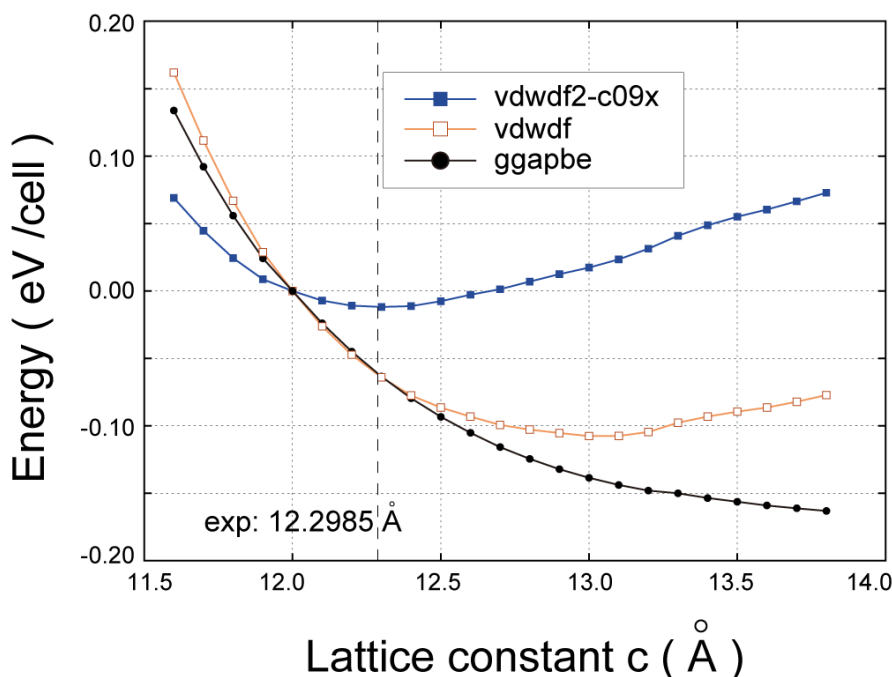


図 5.3-7 積層 2H-MoS₂ の全エネルギーの *c* 軸長依存性

5.3.4.5 制限事項

部分電荷補正がある計算や、PAW の計算には注意してください。内殻電子による補正項が vdwdF に完全対応していないためです。ただし、定性的には大きな影響は与えません。

5.3.4.6 参考文献

- [1] M. Dion, H. Rydberg, E. Schröder, D. C. Langreth, and B. I. Lundqvist: Phys. Rev. Lett. **92** (2004) 246401: Erratum, *ibid*, **95** (2005) 109902.
- [2] O. Gunnarsson and B. I. Lundqvist: Phys. Rev. B **13** (1976) 4274.
- [3] L. X. Benedict, N. G. Chopra, M. L. Cohen, A. Zettl, S. G. Louie, and V. H. Crespi: Chem. Phys. Lett. **286** (1998) 490.
- [4] Y. Baskin and L. Mayer: Phys. Rev. **100**, (1955) 544.
- [5] H. Rydberg, M. Dion, N. Jacobson, E. Schröder, P. Hyldgaard, S. I. Simak, D. C. Langreth, and B. I. Lundqvist: Phys. Rev. Lett. **91** (2003) 126402.
- [6] T. Thonhauser, Valentino R. Cooper, Shen Li, Aaron Puzder, Per Hyldgaard, and David C. Langreth: Phys. Rev. B **76**, 125112 (2007).

5.3.5 ファンデルワールス相互作用補正機能

5.3.5.1 機能の概要

- Williams の方法

R.W. Williams, et al.: Chemical Physics 327 (2006) 54-62

$$E_{vdw} = \sum_{ij} \frac{C_6^{ij}}{R_{ij}^6} f(R_{ij})$$

$$f(R) = \left(1 - \exp \left[-d \left(\frac{R_{ij}}{R_0^{ij}} \right)^7 \right] \right)^4$$

$$C_6^{ij} = -S_C \times \frac{2C_6^i C_6^j p_i p_j}{p_i^2 C_6^i + p_j^2 C_6^j}, \quad R_0^{ij} = S_R \times \frac{(R_0^{ii})^3 + (R_0^{jj})^3}{(R_0^{ii})^2 + (R_0^{jj})^2}, \quad R_0^{ii} = 2 \times R_0^i$$

パラメータ

vdw radius 20.0 bohr

scaling factor S_C 0.8095 (PHASE), S_R 0.80 文献 PBE S_C 0.85 S_R 0.80

damping factor d 3.0

	polarizabilities Å ³	vde coef C6 Hartree*bohr ⁶	vdw radius		polarizabilities Å ³	vde coef C6 Hartree*bohr ⁶	vdw radius
H	0.387	2.831179918	1.17	NTE	0.964	20.89758657	1.50
F	0.296	3.94987377		NTR2	1.030	23.08003267	1.50
Cl	2.315	3.94987377		NPI2	1.090	25.12582491	1.50
Br	3.013	128.2756865		NDI	0.956	20.63799109	1.50
I	5.415	309.0603852		OTE	0.637	11.86370812	1.40
CTE	1.061	22.67403316	1.70	OTR4	0.569	10.01566303	1.40
CTR	1.352	32.61525204	1.70	OPI2	0.274	3.346856941	1.40
CAR	1.352	49.790/Sc	1.70	STE	3.000	121.2531939	1.80
CBR	1.896	54.16430826	1.70	STR4	3.729	168.0350502	1.80
CDI	1.283	30.15058105	1.70	SPI2	2.700	103.5277919	1.80
				PTE	1.538	42.11289383	1.80

- Grimme (DFT-D2)の方法

S. Grimme, J. Comp. Chem. 27, 1787 (2006).

$$E_{disp} = -s_6 \sum_{ij} \frac{C_6^{ij}}{R_{ij}^6} f(R_{ij})$$

$$f(R) = \frac{1}{1 + \exp \left[-d \left(\frac{R_{ij}}{R_0^{ij}} - 1 \right) \right]}$$

$$C_6^{ij} = \sqrt{C_6^i C_6^j}, \quad R_0^{ij} = R_0^i + R_0^j$$

パラメータ

vdw radius 30.0A

scaling factor s_6 0.75, damping factor d 20.0

	C6 Jnm ⁶ /mol	R0 A		C6 Jnm ⁶ /mol	R0 A
H	0.14	1.001	K	10.80	1.485
He	0.08	1.012	Ca	10.80	1.474
Li	1.61	0.825	Sc-Zn	10.80	1.562
Be	1.61	1.408	Ga	16.99	1.650
B	3.13	1.485	Ge	17.10	1.727
C	1.75	1.452	As	16.37	1.760
N	1.23	1.397	Se	12.64	1.771
O	0.70	1.342	Br	12.47	1.749
F	0.75	1.287	Kr	12.01	1.727
Ne	0.63	1.243	Rb	24.67	1.628
Na	5.71	1.144	Sr	24.67	1.606
Mg	5.71	1.364	Y-Cd	24.67	1.639
Al	10.79	1.716	In	37.32	1.672
Si	9.23	1.716	Sn	38.71	1.804
P	7.84	1.705	Sb	38.44	1.881
S	5.57	1.683	Te	31.74	1.892
Cl	5.07	1.639	I	31.50	1.892
Ar	4.61	1.595	Xe	29.99	1.881

1 J/mol = 3.8088e-7 hartree, 1 bohr = 0.5291772480 A

● Grimme (DFT-D3)の方法 (バージョン 2019.01 以上)

DFT-D2 法のほか, DFT-D3 法も利用できます (Stefan Grimme, Jens Antony, Stephan Ehrlich, and Helge Krieg, “A consistent and accurate ab initio parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu” The Journal of Chemical Physics, **132**, 154104 (2010))

DFT-D3 は, DFT-D2 の改良版です。1/ r^8 に比例する項があることやパラメーターが配位数に依存する点などが DFT-D2 と異なる点であり, 特に後者の改良によって原子の局所構造の違いを相互作用に反映させることができるようになっています。

DFT-D3 法を利用するためには, DFT-D2 機能と同じようにまず control ブロックにおいて sw_vdw_correction の値を on にします。

```
Control{
  sw_vdw_correction = on
}
```

さらに, accuracy ブロックにおいて vdw_method として dftd3 を指定すればよい。

```
accuracy{
  vdw_method = dftd3
}
```


DFT-D2 法や Williams の方法と違い、ファンデルワールス相互作用のために別途元素を定義する必要はありません。

また、パラメーターファイルが置いてある場所を file_names.data の F_DFTD3PAR で指定します。

```
&fnames
F_INP = './nfinp.data'
...
F_DFTD3PAR = './dftd3par.data'
/
```

F_DFTD3PAR のデフォルト値は/dftd3par.data です。dftd3par.data ファイルは、PHASE/0 インストールディレクトリーの下以下の場所にあります。

samples/vdw_correction/dftd3

5.3.5.2 入力パラメータ

vdW 補正機能関連のタグ一覧

タグ		値	備考
Control	sw_vdw_correction		
Accuracy	vdw_method	williams grimme or dft-d2	デフォルト
	vdw_radius		20 bohr 30 A (Grimme DFT-D2)
	vdw_scaling_factor		0.805 (Williams) 0.75 (Grimme DFT-D2)
	vdw_scaling_factor_r		0.8 (Williams)
	vdw_damping_factor		3.0 (Williams) 20.0 (Grimme DFT-D2)
Structure	atom_list		
	atoms	#tag vdw で vdw 補正における元素の type を指定	
	vdw_list	vdw 補正における各元素異に対するパラメータ	Williams #tag type c6 r0 p Grimme #tag type c6 r0

赤字は、vdW 補正機能を用いる場合の必須の入力項目

vdW 補正の各元素のパラメータの指定

Williams 法、Grimme(DFT-D2)法の各元素のパラメータをプログラム内で持ち、デフォルト値としている。vdw_list の type は、atom_list で指定した vdw の type と対応している必要がある。

Williams 法

```
vdw_list{
  #tag type c6 r0 p
    H  2.831179918 1.17 0.387
    CTE 22.67403316 1.70 1.061
}
```

Grimme(DFT-D2)法

```
vdw_list{
  #tag type c6 r0
    H  0.14 1.001
```

```

      C      1.75  1.452
    }

```

入力パラメータ例

vdW 補正関連の入力データ例を以下に示す。

Methane Dimer Williams 法

```

Control{
  sw_vdw_correction = ON
}
accuracy{
  vdw_method = williams
  vdw_radius = 20.0
  vdw_scaling_factor = 0.8095
  vdw_scaling_factor_r = 0.8
  vdw_damping_factor = 3.0
}
structure{
  atom_list{
    coordinate_system = cartesian ! {cartesian|internal}
    atoms{
      #units angstrom
      #default mobile=on
      #tag element rx      ry      rz vdw
      C      0      0      0      CTE
      H      0      1.093  0      H
      H      1.030490282 -0.364333333 0      H
      H      -0.515245141 -0.364333333 0.892430763 H
      H      -0.515245141 -0.364333333 -0.892430763 H
      C      0      -3.7  0      CTE
      H      0      -4.793 0      H
      H      -1.030490282 -3.335666667 0      H
      H      0.515245141 -3.335666667 -0.892430763 H
      H      0.515245141 -3.335666667 0.892430763 H
    }
  }
  vdw_list{
    #tag type c6 r0 p
    H      2.831179918 1.17 0.387
    CTE    22.67403316 1.70 1.061
  }
}

```

Methane Dimer Grimme(DFT-D2)法

```

Control{
  sw_vdw_correction = ON
}
accuracy{
  vdw_method = grimme
  vdw_radius = 30.0
  vdw_scaling_factor = 0.75
  vdw_damping_factor = 20.0
}
structure{
  atom_list{
    coordinate_system = cartesian ! {cartesian|internal}
    atoms{
      #units angstrom
      #default mobile=on
      #tag element rx      ry      rz vdw
      C      0      0      0      C
      H      0      1.093  0      H
      H      1.030490282 -0.364333333 0      H
      H      -0.515245141 -0.364333333 0.892430763 H
      H      -0.515245141 -0.364333333 -0.892430763 H
    }
  }
}

```

```

C      0      -3.7    0      C
H      0      -4.793  0      H
H     -1.030490282  -3.335666667  0      H
H      0.515245141  -3.335666667  -0.892430763  H
H      0.515245141  -3.335666667  0.892430763  H
    }
}
vdw_list{
  #tag type c6 r0
      H   0.14 1.001
      C   1.75 1.452
}
}

```

5.3.5.3 計算例

- Water_Dimer (Williams, Grimme(DFT-D2))
- Methane_Dimer (Williams, Grimme(DFT-D2))
- Ethane_Dimer (Williams, Grimme(DFT-D2))
- ATstack (Williams)

5.3.6 有効遮蔽体法 (ESM 法)

5.3.6.1 機能の概要

ESM (Effective Screening Medium)法[1][2]とは、表面モデルを精度よく扱うための計算手法です。PHASEは基底関数として平面波を利用するプログラムなので、厳密には周期系のみ取り扱うことが可能です。表面モデルを扱う場合、表面に垂直な方向に“真空層”を設けることによって表面を模擬した系の計算を行います。このような方法の場合、たとえば分極した表面などは正しく扱えないので特殊な補正を施す必要があります。ESM 法は、実効的な遮蔽物 (effective screening medium) を真空領域に設定することによって半無限におよぶ表面領域を有限の真空領域で扱うことを可能とする方法です。ここでは、PHASE に組み込まれた ESM 法の利用方法を説明します。

5.3.6.2 入力パラメータ

まずは、通常の PHASE 計算と同様の入力パラメータファイルを準備します。この際、原子配置の定義の仕方に注意が必要です。ESM プログラムは、系が以下の図のように定義されていることを仮定しています。

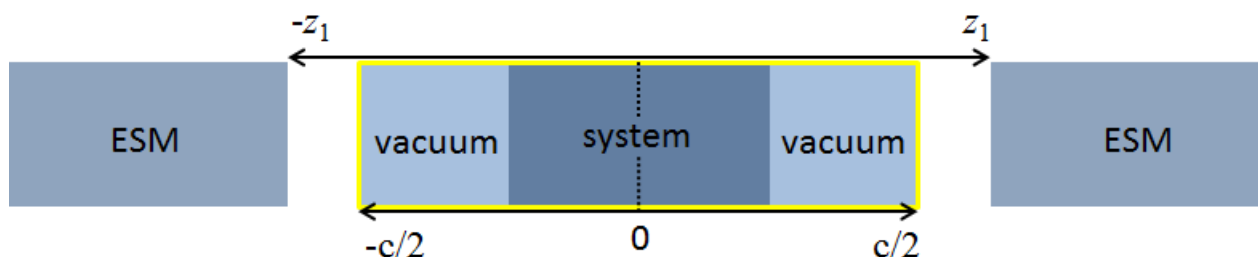


図 5-43 ESM 法において仮定している原子配置の定義方法の模式図

系は、2 方向 (a および b 軸方向) に周期的、1 方向 (c 軸方向) に非周期系であることを仮定しています。c 軸の値が 0 になる場所に系の中心が位置するように必要に応じて系をシフトし、プラスマイナス両側に真空層を設けます。さらに、系の中心から距離 z_1 (入力パラメータファイルによって指定する) 離れた場所に ESM が置かれます。

accuracy ブロックに esm ブロックを作成し、ESM 用の設定を施します。典型的には、以下のとおり。

```
...
...
accuracy{
  esm{
    sw_esm = on
    bc = pe1
    electric_field = 0.001
  }
  ...
  ...
}
...
...
```

esm ブロックの下では、以下の変数を定義することが可能です。

sw_esm	ESM 法を利用するかどうかを指定するスイッチ。on を指定すると ESM 法を利用します。デフォルト値は off
z1	ESM の位置を指定します。指定がない場合、単位胞の境界に ESM が置かれます (すなわち、 $z1=c/2$)。
bc	境界条件(boundary condition)を指定します。BARE, PE1, PE2 のいずれかを指定します。

electric_field	BARE は両側の ESM が真空（誘電率 1）、PE1 は両側の ESM が金属（誘電率 ∞ ）PE2 は ESM の片側が真空、もう片側が金属という境界条件です。デフォルト値は BARE 有限電場を指定したい場合に、その量を原子単位で指定します。この指定は、bc が PE1 であった場合にみに意味をもちます。それ以外の場合、計算中参照されません。電場の単位は、hartree/bohr です（約 51.4 V/Å）。
add_elec	電子を追加/削除したい場合に追加/削除したい電子数を実数で指定します。削除したい場合は負の数を指定します。
z_wall	原子が真空層のある領域からはみでないように“壁”を設ける場合に、その“ある領域”の指定を実数で行います。このパラメータの指定があった場合にこの機能が有効となります。
bar_width	“壁” ポテンシャルを設ける場合に、その幅を長さの単位で指定します。
bar_height	“壁” ポテンシャルを設ける場合に、その高さをエネルギーの単位で指定します。

5.3.6.3 計算の実行

計算の実行は、通常の PHASE の計算通り行います。並列の方法や擬ポテンシャルの制限などは特にありません。利用できる計算機能にも特に制約はありません。

5.3.6.4 計算例

（1）水分子

ESM 法適用例として、単純な水分子の計算結果を紹介します。この例題の入力ファイルは、samples/esm/H2O 以下にあります。水分子は、単体で双極子モーメントをもっているので、水分子は通常の周期系の計算の場合全エネルギーに有限の誤差が発生します。ESM 法は、1つの方向については無限の計算を行うことになるので、正しい全エネルギーが得られると考えられます。

このようなことを確認するため、以下のような計算を実施します。

1. 水分子単体の、周期系における全エネルギー計算
2. 水分子単体の、ESM 法による全エネルギー計算
3. 水分子を互い違いに配置した系の、周期系における全エネルギー計算

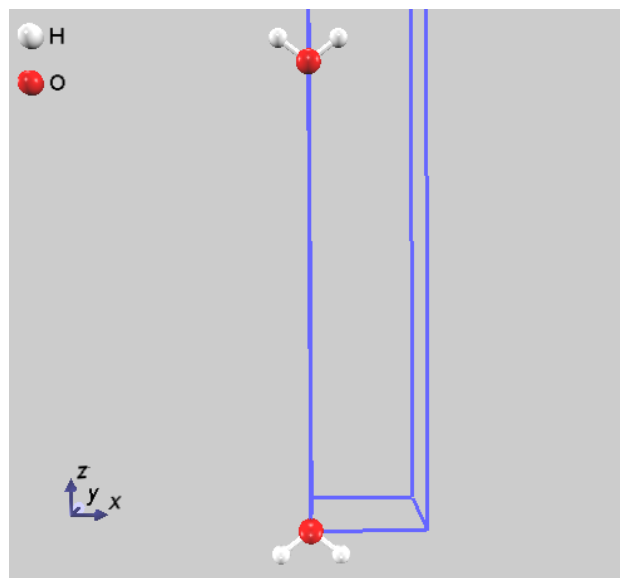


図 5-44 H₂O 分子を互い違いに配置した系。

3. のように水分子を互い違いに配置した系を用意することによって、双極子モーメントを打ち消すことが可能です。したがって、3.の計算によって得られた全エネルギーの半分の値は、2. の計算と結果が一致するはずです。ここでは、このような結果が得られるかどうかを確認します。

計算された全エネルギーの結果は、以下に示します。

	全エネルギー (hartree/H ₂ O)	参照値との差 (hartree/H ₂ O)
1.の計算	-17.1855148193	$-1.8927504 \times 10^{-3}$
2.の計算	-17.1836307637	8.6948×10^{-6}
3.の計算	-17.1836220689	0

表から明らかなように、ESM 法による計算と 3. の計算の結果は非常に近く、“無限の系の計算”が ESM 法によって実現できていることがわかります。

(2) 電場を印加した計算例

ESM 法を利用すると、*c* 軸方向に電場を印加した計算を実施することが可能です。ここでは、簡単な例によってその利用方法を説明します。この例題の入力ファイルは `samples/esm/Al111` 以下にあります。

採用した系は、仕事関数の計算例でも利用した Al(111) の系です。以下のような設定によって ESM を利用します。

```
accuracy{
  esm{
    sw_esm = on
    bc = pe1
    electric_field = 0.001
  }
}
```

電場を印加するには、パラメータ `bc` として `pe1` を指定する必要がある点に注意してください。また、電場の影響をみるために局所ポテンシャルを出力するので、仕事関数に関する設定も有効にしています。

```
postprocessing{
  workfunc{
    sw_workfunc = on
  }
}
```

電場の大きさは、-0.001, 0, +0.001 (単位は原子単位) と変化させて計算を行いました (0 のケースは通常の計算とほぼ同じです)。最後に、この系には反転対称性がありますが、電場を印加すると反転対称性は損なわれるので `sw_inversion` パラメータは指定していません。

以上の設定のもと、通常通り PHASE を実行します。各ケース計算が終了したら、`workfunc` プログラムを利用してポスト処理を実施します。この処理の結果得られる局所ポテンシャルの、電場を印加する場合としない場合の差と表面に垂直な軸方向の距離の関係を図 5-45 に示します。この図においては、距離のちょうど半分の地点が系の境、すなわち図 5-45 の $\pm c/2$ の地点になっています。

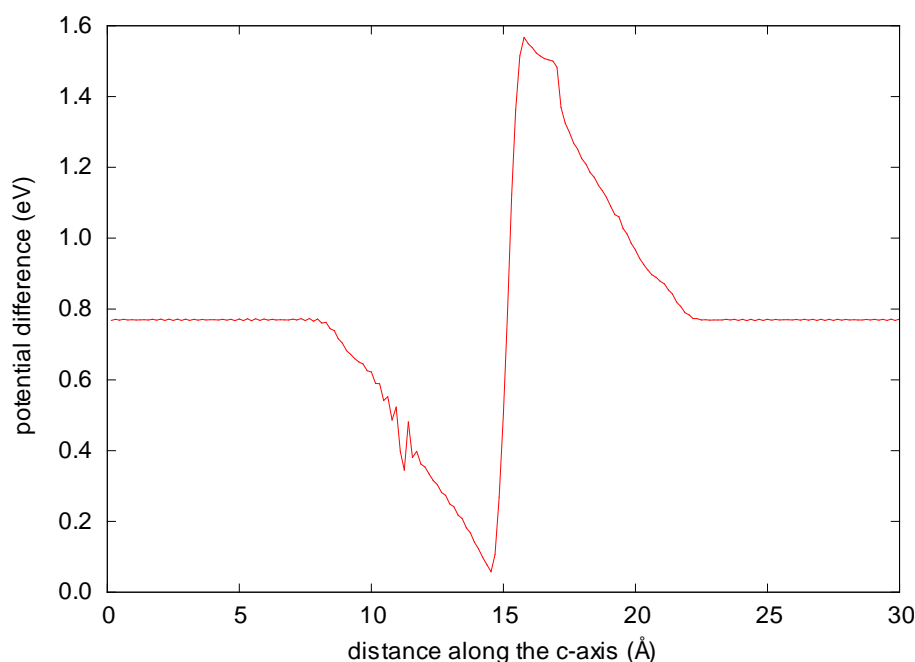


図 5-45 ローカルポテンシャルの差分と表面に垂直な距離との関係。

図から明らかなように、電場を印加することによって真空領域の局所ポテンシャルにかたむきが発生しています。また、金属域においてはフラットとなっています。これは、電場は真空域で発生し、金属中では発生していないことを表わしています。

5.3.6.5 使用における注意点

- ESM 法を利用する場合、コンパイルの際にフーリエ変換ライブラリーとして **FFTW** を利用する必要があります。
- 原子配置の指定においては、反転対称の位置の原子は直接指定するようにし、**weight** パラメータはつねに 1 としてください。
- 電場を印加する場合 *c* 軸方向の対称性はあったとしても損なわれるので、そのような対称性は無効にするようにしてください。また、**sw_inversion** パラメータを **on** にはしないでください。

5.3.6.6 参考文献

- [1] M. Otani and O. Sugino, “First-principles calculations of charged surfaces and interfaces: A plane-wave nonrepeated slab approach”, *Physical Review B* **73**, 115407 (2006).
- [2] Hamada, M. Otani, O. Sugino and Y. Morikawa, “Green’s function method for elimination of the spurious multipole interaction in the surface/interface slab model”, *Physical Review B* **80**, 165411 (2009).

5.3.6.7 ライセンス

PHASE の ESM 機能は、EsmPack ライブラリーを通じて実現されています。EsmPack ライブラリーは、産業技術総合研究所から MIT ライセンスで公開されている、ESM 法を実現するための汎用のライブラリーです。以下のライセンス条項をご理解いただいた上でご利用ください。また、本機能を利用して論文発表などを行う場合、必ず文献[1], [2]を参考文献に含めてください。

Copyright (c) 2012, Minoru Otani <minoru.otani@aist.go.jp>
 Permission is hereby granted, free of charge, to any person

obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

5.3.7 Dipole 補正

Dipole 補正機能とは、系が荷電状態の場合にその影響を打ち消す **dipole** を配置することによって補正し、正しい全エネルギーを求めたり、電場を印加した計算を行うことができる機能です。Dipole の大きさはセルフコンシステントに決まるので、ユーザーが適切な **dipole** の大きさを勘案する必要はありません。また、原子間力計算においても考慮されるので、構造最適化に利用することも可能となっています。

5.3.7.1 入力パラメーターファイルの書き方

Dipole 補正機能を有効にするには、入力パラメーターファイルの **control** ブロックにおいて以下のような記述を行います。

```
control{
  sw_dipole_correction = on
}
```

この指定によって、**dipole** 補正機能が有効になります。さらに、**accuracy** の **dipole** ブロックにおいて、**dipole** 補正機能の詳細設定を行います。

```
accuracy
...
...
dipole_correction{
  direction = 3
  amix = 1
  vacuum{
    rz = -0.5
  }
  elec_field{
    ez = 0.0
  }
}
```

dipole_correction の下の **vacuum** ブロックで **dipole** を配置する位置を、**elec_field** ブロックで外部電場を設定することができます。**dipole_correction** ブロックにおいて利用できるパラメーターは次の表に示す通りです。

パラメーター名	説明
direction	双極子の向きを指定します。1 が <i>a</i> 軸方向, 2 が <i>b</i> 軸方向, 3 が <i>c</i> 軸方向に対応します。
amix	双極子のミクシングパラメーターです。0 よりも大きく, 1 以下の値を指定します。デフォルト値は 1 です。
vacuum	双極子位置を指定するブロックです。双極子の位置は, 単位胞の長さを 1 とする単位で指定します。
rx	双極子の, <i>a</i> 軸方向の位置です。 direction = 1 の場合に設定します。双極子は, 真空中に配置します。
ry	双極子の, <i>b</i> 軸方向の位置です。 direction = 2 の場合に設定します。双極子は, 真空中に配置します。
rz	双極子の, <i>c</i> 軸方向の位置です。 direction = 3 の

		場合に設定します。双極子は、真空中に配置します。
elec_field		印加したい外部電場を指定するブロックです。
	ex	電場の、 a 軸方向の値です。direction = 1 の場合に設定します。
	ey	電場の、 b 軸方向の値です。direction = 2 の場合に設定します。
	ez	電場の、 c 軸方向の値です。direction = 3 の場合に設定します。

Dipole の位置の指定方法には制限があるので、注意して指定する必要があります。双極子の向きを z 方向とすると、「rz+0.5 の位置に系の中心が位置するように」、そして「周期境界条件は考慮せずに」指定します。たとえば、図 5-46 のように単位胞の中心に系を置く場合、rz = 0 とします。周期境界条件下では rz = 1 としてもよさそうですが、このような指定は不正となります。

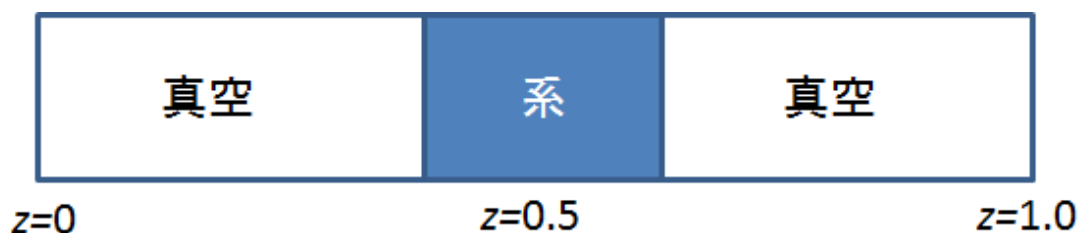


図 5-46 Dipole 補正を有効にする表面系の模式図

また、dipole 補正機能と直接は関係ありませんが、下記のような設定を行うことによって局所ポテンシャルを出力することができます。この設定を施した上で仕事関数を求める際の手続きを踏むことによって、真空層の方向と局所ポテンシャルの関係を解析することが可能です。

```
postprocessing{
  workfunc{
    sw_workfunc = on
  }
}
```

5.3.7.2 ログファイルの見方

Dipole 補正を有効にした状態で計算を実行すると、outputxxx ファイルに以下のように dipole の履歴が SCF ステップごとに出力されます。

```
Dipole: rmin,rmax,idir =   -0.50000000    0.50000000  3
Dipole: Total =   -0.22795296 =   -0.5795 Debye =   -1.9326x10^-30 Cm
Dipole: Ion   =   -38.10003241
Dipole: Elec  =   -37.87207945
Dipole: mix dipole and field with amix_dip=    1.00000000
Dipole: (NOW) dipole,field =   -0.00776990    0.00000000
Dipole: (OLD) dipole,field =   -0.00776992    0.00000000
Dipole: (NEW) dipole,field =   -0.00776990    0.00000000
Dipole: Edip(ion),Eext(ion)=    0.01230    0.00000
Dipole: potential jump (dip and ext)=   -0.09764    0.00000
```

Dipole: Total で報告されている値は dipole の大きさそのものです。Dipole: (NOW), Dipole: (OLD), Dipole: (NEW)

で報告されている値はそれぞれ現ステップの **dipole** の値, 1 ステップ前の **dipole** の値, ミクシングの結果得られた **dipole** 値に相当しますが, SCF に足しこめるポテンシャルになるよう体積などで規格化された量となっているため値の大きさは **Dipole: Total** のそれと異なります。

5.3.7.3 計算例

Dipole 補正を利用した計算例として, ESM と同様水分子の例と電場を印加した計算例を挙げます。

(1) 水分子

ESM の例 (5.3.6.4 (1)) と同じセッティングで, Dipole 補正を利用した全エネルギー計算を行ってみました。この例において, Dipole は以下のように設定しています。

```
control{
  sw_dipole = on
}
accuracy{
  ...
  dipole_correction{
    amix = 1
    direction = 3
    vacuum{
      rz = -0.5
    }
  }
}
...

```

単位胞の一番下に水分子を配置しているので, **vacuum** ブロックの変数 **rz** の値は-0.5 となっています。結果は次の表に示す通りです (水分子の配向を少し変えたので, ESM の例と全く同じではありません)。

表 3 水分子の結果の比較

	補正なし	補正あり	esm	ref
エネルギー(ha)	-17.201981	-17.201719	-17.2017015	-17.201773
差 (ha)	-0.000208	0.000054	0.000072	—

ESM の場合と同様, 補正しない場合と比較してより参照値に近いエネルギーを得ることができました。

図 5-47 に, *c* 軸方向と局所ポテンシャルの関係を補正なし, 補正あり, ESM 法の場合についてプロットしました。この図においては, *c* 軸が 0 および 16Å 近辺の位置に水分子が配置されています。補正なしの場合, 真空層においても局所ポテンシャルが傾きを持っています。これは, 周期的に並んだ水分子がクーロン相互作用によって相互作用していることを意味しています。他方, Dipole 補正の場合と ESM 法の場合は真空のちょうど中心において局所ポテンシャルに飛びが現れています。両手法の場合水分子はお互いに相互作用を持たず, その影響を真空層の中心あたりで吸収していることを意味しています。

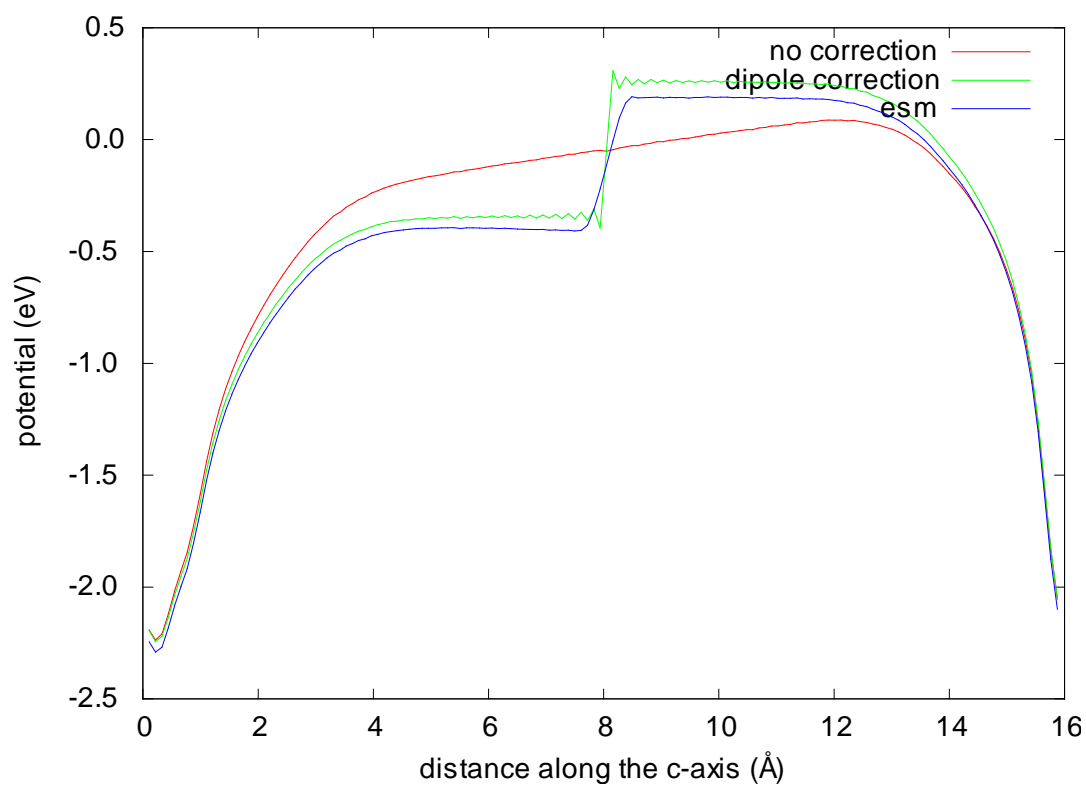


図 5-47 真空層と局所ポテンシャルの関係

(2) 電場を印加した計算

やはり **ESM** の例 (5.3.6.4 (1)) と同じセッティングで, **Dipole** 補正機能を利用して電場を印加した計算を行いました。その設定は, 以下のようにになっています。

```
control{
  sw_dipole = on
}
accuracy{
  ...
  dipole_correction{
    amix = 1
    direction = 3
    vacuum{
      rz = -0.5
    }
    electric_field{
      ez = 0.001
    }
  }
}
...
postprocessing{
  workfunc{
    sw_workfunc = on
  }
}
```

以上の設定で、通常通り PHASE を実行し、終了後 workfunc プログラムを利用してポスト処理を実施します。この処理の結果得られる局所ポテンシャルの、電場を印加する場合としない場合の差と表面に垂直な軸方向の距離の関係を図 5-48 に示します。

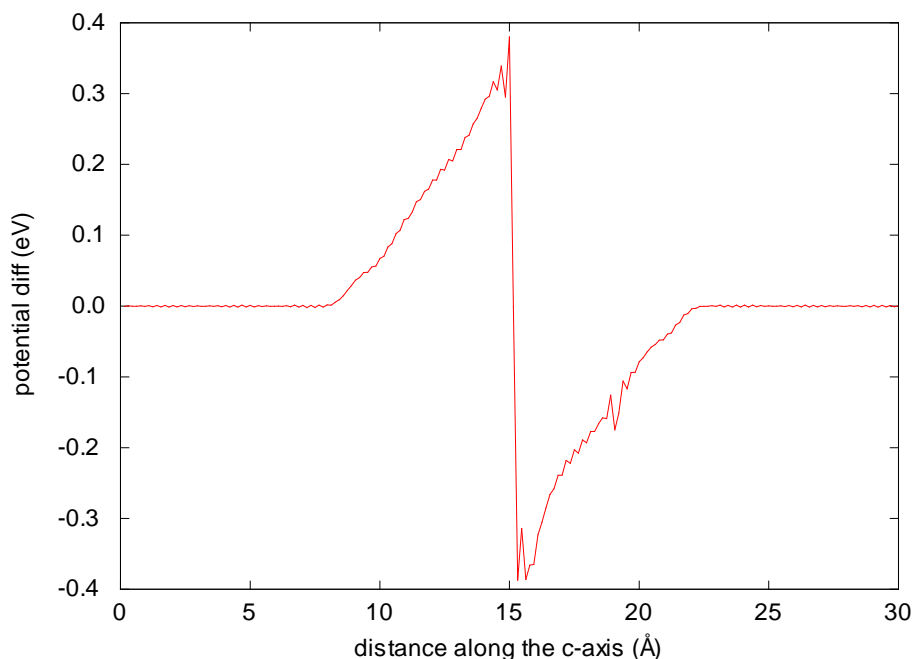


図 5-48 ローカルポテンシャルの差分と表面に垂直な距離との関係。

ESM の場合と同様、ポテンシャルの差分は、真空域においては傾きがあり、金属中においてはフラットになりました。これは金属中では電位差が発生していないことを表しており、基本的な電磁気学の知見と合う結果となっています。

5.4 化学反応の解析

5.4.1 NEB 法

5.4.1.1 機能の概要

Nudged Elastic Band (NEB)法および Climbing Image (CI) NEB 法は、反応経路における始状態と終状態の間の最小エネルギー経路と鞍点を求める方法です。

NEB 法および CI-NEB 法を用いた反応経路計算においては、始状態の原子配置(\vec{R}_0)および終状態の原子配置(\vec{R}_N)

が既知であるとして、始状態と終状態の間の中間状態(\vec{R}_i , $i = 2 \sim N - 1$)の原子配置やエネルギーなどを、隣接

する状態(イメージ)間がばねによって結ばれているという拘束条件の下で構造最適化計算を行います。ここで \vec{R}_i は、各イメージにおける $3M$ 次元(M は原子数)の座標です。NEB 法および CI-NEB 法の中間状態における初期原子配置は始状態と終状態から任意に決定することが可能ですが、始状態と終状態をイメージで等分割し決める方法が多くの場合採用されます。

● NEB 法

通常の NEB 法においては、各イメージの作用力は次のように与えられます。

$$\vec{F}_i = \vec{F}_i^s|_{\parallel} - \nabla E(\vec{R}_i)|_{\perp}.$$

ここで、 $\vec{F}_i^s|_{\parallel}$ は隣接するイメージ間のばねによる作用力の接線方向の成分であり、以下のように求められます。

$$\vec{F}_i^s|_{\parallel} = k(|\vec{R}_{i+1} - \vec{R}_i| - |\vec{R}_i - \vec{R}_{i-1}|) \cdot \hat{t}\hat{t}.$$

ここで k はばね定数です。 \hat{t} は接線方向の単位ベクトルであり、次のように計算します。

$$\hat{t} = \frac{\vec{R}_i - \vec{R}_{i-1}}{|\vec{R}_i - \vec{R}_{i-1}|} + \frac{\vec{R}_{i+1} - \vec{R}_i}{|\vec{R}_{i+1} - \vec{R}_i|}.$$

(11)式の $\nabla E(\vec{R}_i)|_{\perp}$ は、第一原理計算などによって得られる、原子に働く力の接線に垂直な成分であり、以下のよう求められます。

$$\nabla E(\vec{R}_i)|_{\perp} = \nabla E(\vec{R}_i) - \nabla E(\vec{R}_i) \cdot \hat{t}\hat{t}.$$

● CI-NEB 法

CI-NEB 法は、通常の NEB 法に対して最もエネルギーの高いイメージにおける作用力の計算方法を改良した方法です。CI-NEB 法計算では、通常の NEB 法を用いて反応経路計算をある程度進めた後に最も高いエネルギーのイメージ($\vec{R}_{i,\max}$)を決定し、 $\vec{R}_{i,\max}$ に働く作用力を次のように計算します。

$$\begin{aligned}\vec{F}_{i,\max} &= -\nabla E(\vec{R}_{i,\max}) + 2\nabla E(\vec{R}_{i,\max}) \\ &= \vec{F}_{i,\max}|_{\perp} - \vec{F}_{i,\max}|_{\parallel}\end{aligned}$$

● ばね定数の計算方法

最少遷移エネルギーを求める反応経路計算においては、鞍点付近の計算精度を高くすることが好ましいと考えられます。このことから、鞍点付近でイメージの密度を高くし、接線の傾きを高い精度で求める必要があります。特に、反応経路全体と比較してポテンシャル障壁の領域が極めて狭い場合には、ポテンシャル障壁近傍のイメージ密度を高くすることにより高精度の計算を効率よく行うことが可能となります。最少エネルギー経路において鞍点付近にイメージを密に分布させる方法として、鞍点付近のばね定数 k を大きくする方法が考えられています。NEB 法および CI-NEB 法における作用力は、ばねによる作用力とエネルギー計算によって得られた作用力の線形結合で表わされるので、イメージ間のばね定数は異なる値を選択することが可能です。ばね定数の設定方法としては、以下のエネルギーの線形関数が提唱されています。

$$k = k_{\max} - \Delta \left(\frac{E_{\max} - E_i}{E_{\max} - E_{\text{ref}}} \right) \quad (E_i \geq E_{\text{ref}} \text{ の場合}),$$

$$k = k_{\max} - \Delta k \quad (E_i < E_{\text{ref}} \text{ の場合}).$$

ここで、 k_{\max} はばね定数の最大値、 Δk はばね定数の最大値と最小値の差です。 E_i は i 番目のばねで結ばれた2つのイメージのうち高いエネルギーのイメージのエネルギー、 E_{\max} は全イメージ中最も高いエネルギー、 E_{ref} は始状態と終状態のうち、高い方のエネルギーです。この E_{ref} の値の設定によって、反応経路における始状態付近と終状態付近のイメージ密度が等しくなります。

5.4.1.2 入力パラメータ

(1) 入力パラメータの指定

NEB 法に関連する、入力データのタグおよびその説明を以下に示します。

NEB に関連する入力データ

第 1 ブロック	第 2, 第 3 ブロック	タグ識別子	説明
Control			
		multiple_replica_mode	NEB 計算の実行 ON,OFF
		multiple_replica_max_iteration	NEB iteration 数
multiple_replica			
		method	反応経路計算手法 nudged_elastic_band_method
	accuracy		
		dt	NEB 計算における原子座標更新の Δt
		neb_time_integral	時間積分法 quench, steepest_descent(デフォルト)
		penalty_function	ペナルティ関数 ON,OFF
		neb_convergence_condition	NEB 収束判定法(後述)
		neb_convergence_threshold	NEB 収束判定値
	constraint		
		ci_neb	CI-NEB ON, OFF
		sp_k_init	ばね定数(初期値)
		sp_k_min	ばね定数(最小値)
		sp_k_max	ばね定数(最大値)
		sp_k_variable	ばね定数の固定, 変動 OFF 固定, ON 変動
	structur		
		number_of_replicas	レプリカ数
	replica		レプリカ情報
		endpoint_images	両端のイメージの指定方法 directin, file
	atom_list_end0		両端のイメージの原子リスト
	atom_list_end1		両端のイメージの原子リスト
structure			
	symmetry		
		method	対称性の与え方

NEB 法の入力パラメータの指定について説明します。

NEB 法の計算は、以下の指定を行います。

- NEB法の機能を有効にする
- NEB用の収束判定の設定を行う
- 結晶の対称性を設定する
- レプリカ列両端のレプリカの座標データを設定する
- レプリカの間レプリカの座標データを設定する

各々について、以下に説明をします。

- NEB 機能を有効にする。

PHASE に NEB 法による計算を実行することを伝えるため、以下のように control ブロックの下で multiple_replica_mode 変数を on とします。

```
control{
```



```
multiple_replica_mode = on
}
```

- 収束判定

収束判定条件は、`multiple_replica` ブロックの下に `accuracy` ブロックの変数 `neb_convergence_condition` で設定します。

```
multiple_replica{
  accuracy{
    neb_convergence_condition = energy_e
  }
}
```

`neb_convergence_condition` には、数値または文字列を指定することができます。表 12 に、設定できる条件を示します。

収束判定条件設定のパラメータ

数値	文字列	説明
1	<code>energy_e</code>	<code>dE < threshold</code>
2	<code>phase_force</code>	PHASE の力の最大値 <code>< threshold</code>
3	<code>neb_force</code>	NEB 計算で補正した力の最大値 <code>< threshold</code>
4	<code>force_at_transition_state</code>	最大エネルギーイメージの PHASE の力の最大値 <code>< threshold</code>
5	<code>phase_force_normal</code>	PHASE の、経路に垂直な成分の力の最大値 <code>< threshold</code>

- 結晶の対称性を指定する

対称性に自動探索 (`structure{symmetry{method = automatic}}`) を設定すると、両端のレプリカのいずれかあるいは両方に、中間のレプリカよりも高い対称性が見つかった場合に、正常に計算が進まなくなります。マニュアル設定にして、必要ならば生成元を与えることによって (4.2.1.2 参照)、全レプリカの対称性を揃えて下さい。

- 両端のレプリカの指定：入力で直接指定する方法

両端のレプリカの原子座標を入力において直接指定するには、以下のような記述を行います。

```
multiple_replica{
  ....
  ....
  structure{
    ....
    ....
    endpoint_images = directin
    atom_list_end0{
      coordinate_system = cartesian ! {internal|cartesian}
      atoms{
        #units angstrom
        #tag element rx ry rz
        Si 0.000000000000 0.000000000000 0.000000000000
        Si 2.751721694800 2.751721694800 0.000000000000
        ....
        ....
      }
    }
    atom_list_end1{
      coordinate_system = cartesian ! {internal|cartesian}
      atoms{
        #units angstrom
        #tag element rx ry rz
        Si 0.000000000000 0.000000000000 0.000000000000
```

```

        Si      2.751721694800      2.751721694800      0.000000000000
        ....
        ....
    }
}
    ....
    ....
}
    ....
    ....

```

変数 `endpoint_images` に `directin` という文字列を指定し、さらに `atom_list_end0` ブロックに始状態の、`atom_list_end1` に終状態のレプリカの座標値を通常の **PHASE** の `atom_list` ブロックにおける指定と同じように指定します。

- 両端のレプリカの指定：両端のレプリカの原子座標をファイルから指定する方法

両端のイメージの原子座標をファイルで指定する場合は、入力データの `endpoint_images` の値を `file` とし、`file_names.data` にイメージのファイル名を設定します。その際、`file_names.data` ファイル中では `F_IMAGE(-1)` および `F_IMAGE(0)` というファイルポインターを利用します。以下は、入力データと `file_names.data` の記述例です。

入力データの記述例

```

multiple_replica{
    ...
    ...
    structure{
        endpoint_images = file
    }
    ...
    ...
}

```

`file_names.data` ファイルの記述例

```

&fnames
...
...
/
&nebfiles
F_IMAGE(0) = './endpoint0.data'
F_IMAGE(-1) = './endpoint1.data'
...
...
/

```

また、原子座標データファイル(上記の例では `endpoint0.data` や `endpoint1.data` というファイル名)は、次のような形式で記述します。

```

coordinate_system=cartesian

#units angstrom

Si      0.000000000000      0.000000000000      0.000000000000
Si      2.751721694800      2.751721694800      0.000000000000
Si      1.375860847400      1.375860847400      1.375860847400
Si      4.127582542200      4.127582542200      1.375860847400
Si      0.000000000000      2.751721694800      2.751721694800
Si      2.751721694800      0.000000000000      2.751721694800

```

Si	1.375860847400	4.127582542200	4.127582542200
Si	4.127582542200	1.375860847400	4.127582542200
Si	0.000000000000	0.000000000000	5.503443389600
Si	2.751721694800	2.751721694800	5.503443389600
Si	1.375860847400	1.375860847400	6.879304237000
H	1.644706293661	1.095414892118	11.000000000000
H	1.095414929519	1.644706317263	11.000000000000

- 中間レプリカの指定：中間レプリカの原子座標を両端の原子座標の線形補間で指定する方法 (proportional)

中間レプリカの原子座標を両端の原子座標の線形補間で指定する場合は、replica タグ内の howtogive_coordinates を **proportional** とします。入力データの記述例を以下に示します。

```
multiple_replica{
  structure{
    number_of_replicas = 6
    replicas{
      #tag replica_number howtogive_coordinates end0 end1
      1 proportional 0 -1 ! 0: end0, -1:end1
      2 proportional 0 -1
      3 proportional 0 -1
      4 proportional 0 -1
      5 proportional 0 -1
      6 proportional 0 -1
    }
  }
}
```

- 中間レプリカの指定：中間レプリカの原子座標をファイルから指定する方法(file)

中間イメージをファイルで指定する場合は、replica タグ内の howtogive_coordinates を **file** とし、対応する原子座標ファイルは file_names.data ファイルで指定します。入力データと file_names.data ファイルの記述例を以下に示します。

入力データの記述例

```
multiple_replica{
  ...
  ...
  structure{
    number_of_replicas = 3
    replicas{
      #tag replica_number howtogive_coordinates end0 end1
      1 file 0 -1 ! 0: end0, -1:end1
      2 file 0 -1
      3 file 0 -1
    }
  }
}
```

file_names.data の記述例

```
&fnames
...
...
/
&nebfles
F_IMAGE(0) = './endpoint0.data'
F_IMAGE(-1) = './endpoint1.data'
```

```

F_IMAGE(1) = './image1.data'
F_IMAGE(2) = './image2.data'
F_IMAGE(3) = './image3.data'

/

```

原子座標データを指定するファイルの書式は、両端イメージの場合と同じです。

入力パラメータの例を以下に示します。

```

Control{
  condition = initial ! {initial|continuation|automatic}
  cpumax = 1 day ! {sec|min|hour|day}
  max_iteration = 10000000
  multiple_replica_mode = ON
  multiple_replica_max_iteration = 2000
}
accuracy{
  cutoff_wf = 10.00 rydberg
  cutoff_cd = 40.00 rydberg
  num_bands = 28
  ksampling{
    method = monk ! {mesh|file|directin|gamma}
    mesh{ nx = 2, ny = 2, nz = 1 }
  }
  smearing{
    method = parabolic ! {parabolic|tetrahedral}
    width = 0.001 hartree
  }
  xctype = ggapbe
  scf_convergence{
    delta_total_energy = 0.5e-7 hartree
    succession = 2 !default value = 3
  }
  initial_wavefunctions = matrix_diagon !{random_numbers|matrix_diagon}
  matrix_diagon{
    cutoff_wf = 3.00 hartree
  }
}
structure{
  unit_cell_type = primitive
  unit_cell{
    a_vector = 10.400 0.000 0.000
    b_vector = 0.000 10.400 0.000
    c_vector = 0.000 0.000 30.200
  }
  symmetry{
    method = manual
    sw_inversion = off
  }
  atom_list{
    coordinate_system = cartesian ! {cartesian|internal}
    atoms{
      #units angstrom
      #tag element rx ry rz mobile
Si 0.000000000000 0.000000000000 0.000000000000 0
Si 2.751721694800 2.751721694800 0.000000000000 0
Si 1.375860847400 1.375860847400 1.375860847400 0
Si 4.127582542200 4.127582542200 1.375860847400 0

```

```

Si      0.000000000000      2.751721694800      2.751721694800      0
Si      2.751721694800      0.000000000000      2.751721694800      0
Si      1.375860847400      4.127582542200      4.127582542200      0
Si      4.127582542200      1.375860847400      4.127582542200      0
Si      0.000000000000      0.000000000000      5.503443389600      0
Si      2.751721694800      2.751721694800      5.503443389600      0
Si      1.375860847400      1.375860847400      6.879304237000      0
H       1.644706293661      1.095414892118      11.000000000000      1
H       1.095414929519      1.644706317263      11.000000000000      1
    }
}
element_list{
    #tag element  atomicnumber  mass  zeta  dev
    #units atomic_mass
    Si      14      28.085
    H       1       1.008
}
}
multiple_replica{
    method = nudged_elastic_band_method
    accuracy{
        dt = 40 au_time
        neb_time_integral = quench
        penalty_function = off
        neb_convergence_condition = 3
        neb_convergence_threshold = 5.0e-04
    }
    constraint{
        ci_neb = OFF
        sp_k_init = 0.03
        sp_k_min = 0.03
        sp_k_max = 0.03
        sp_k_variable = OFF
    }
    structure{
        number_of_replicas = 6
        replicas{
            #tag replica_number  howtogive_coordinates  end0  endl
            1      proportional      0      -1 ! 0: end0, -1:endl
            2      proportional      0      -1
            3      proportional      0      -1
            4      proportional      0      -1
            5      proportional      0      -1
            6      proportional      0      -1
        }
        endpoint_images = directin ! {no or nothing | file | directin}
        howtogive_coordinates = from_endpoint_images
        atom_list_end0{
            coordinate_system = cartesian ! {internal|cartesian}
            atoms{
                #units angstrom
                #tag element rx ry rz
                Si      0.000000000000      0.000000000000      0.000000000000
                Si      2.751721694800      2.751721694800      0.000000000000
                Si      1.375860847400      1.375860847400      1.375860847400
                Si      4.127582542200      4.127582542200      1.375860847400
                Si      0.000000000000      2.751721694800      2.751721694800
                Si      2.751721694800      0.000000000000      2.751721694800
                Si      1.375860847400      4.127582542200      4.127582542200
                Si      4.127582542200      1.375860847400      4.127582542200
            }
        }
    }
}

```

```

        Si    0.000000000000    0.000000000000    5.503443389600
        Si    2.751721694800    2.751721694800    5.503443389600
        Si    1.375860847400    1.375860847400    6.879304237000
        H     1.644706293661    1.095414892118    11.000000000000
        H     1.095414929519    1.644706317263    11.000000000000
    }
}
atom_list_end1{
    coordinate_system = cartesian ! {internal|cartesian}
    atoms{
        #units angstrom
        #tag element rx ry rz
        Si    0.000000000000    0.000000000000    0.000000000000
        Si    2.751721694800    2.751721694800    0.000000000000
        Si    1.375860847400    1.375860847400    1.375860847400
        Si    4.127582542200    4.127582542200    1.375860847400
        Si    0.000000000000    2.751721694800    2.751721694800
        Si    2.751721694800    0.000000000000    2.751721694800
        Si    1.375860847400    4.127582542200    4.127582542200
        Si    4.127582542200    1.375860847400    4.127582542200
        Si    0.000000000000    0.000000000000    5.503443389600
        Si    2.751721694800    2.751721694800    5.503443389600
        Si    1.375860847400    1.375860847400    6.879304237000
        H     2.22686927    0.48813212    7.65400988
        H     0.48813224    2.22686933    7.65400957
    }
}
}
}
wavefunction_solver{
    solvers{
        #tag sol till_n dts dte itr var prec cmix submat
        lmMSD -1 0.2 0.2 1 linear on 1 on
    }
}
charge_mixing{
    mixing_methods{
        #tag no method rmxs rmxe itr var prec istr nbmix update
        1 broyden2 0.10 0.10 1 linear on 1 0 RENEW
    }
}
printoutlevel{
    base=1
}
}

```

(2) NEB 関連ファイルの指定

NEB 関連のファイルは, `file_names.data` で設定します。次のように記述します。

```

&fnames
F_INP='./nfinp.data'
F_POT(1)='./Si_ggapbe_nc_01.pp'
...
...
/
&nebfiles
F_IMAGE(0) = './endpoint0.data'

```

```

F_IMAGE(-1) = './endpoint1.data'
F_NEB_OUT   = './output_neb'
F_NEB_ENF   = './nfnebenf.data'
F_NEB_DYNM  = './nfnebdynm.data'
/

```

ファイル読み込みの **namelist** として、&nebfiles を利用している点にご注意ください。
 &nebfiles で利用できるファイルポインターを、表 13 に示します。

NEB で利用できるファイルポインター

ファイル名変数	Unit 番号	デフォルト値	備考
F_IMAGE(-1:99)	201	./endpoint0.data (F_IMAGE(0)) ./endpoint1.data (F_IMAGE(1))	イメージの原子座標
F_NEB_STOP	202	./nfnebstop.data	NEB ステップ終了用ファイル
F_NEB_OUT	203	./output_neb	NEB 計算 ログ出力
F_NEB_CNTN	204	./neb_continue.data	NEB 継続計算用ファイル
F_NEB_ENF	205	./nfnebenf.data	エネルギー、力出力ファイル
F_NEB_DYNM	206	./nfnebdynm.data	原子座標出力ファイル

5.4.1.3 計算の実行方法

NEB は「レプリカ並列」に対応しています。以下のように起動します。

```
% mpirun -n NP phase ne=NE nk=NK nr=NR
```

ここで、NP は MPI プロセスの数、NR は並列で計算するレプリカの数、NE、NK は PHASE と同様バンドおよび **k** 点並列の数です。ただし、NP = NR × NE × NK という関係が成立している必要があります。

5.4.1.4 計算結果の出力

NEB シミュレーションを実行すると、通常の PHASE の計算と比較して多くのファイルが得られます。まず、ログファイル(output000)や継続計算に利用されるファイル(continue.data ファイルなど)は すべてレプリカ毎に出力されます。識別のため、それぞれのファイルの末尾に“_rxxx”という文字列がたされます。さらに NEB 固有の以下のファイルが得られます。

- output_neb_pxxx

NEB 計算のログファイルです。xxx には MPI プロセスの番号が割り振られます。NEB 計算に関するログが出力されます。

- nfnebenf.data

NEB 計算のエネルギーや NEB 力などが記録されたファイルです。以下のような形式で出力されます。

#step	image	image_distance	energy	force_org	force_neb	force_normal
1	1	0.0000000000E+00	-0.4399458479E+02	0.1112676571E-01	0.1112676571E-01	0.0000000000E+00
1	2	0.1323772380E+01	-0.4397221867E+02	0.5212041989E-01	0.4899393390E-01	0.4899393390E-01
1	3	0.2640972887E+01	-0.4393533860E+02	0.5368141337E-01	0.5023308254E-01	0.5023308254E-01
1	4	0.3958252743E+01	-0.4389613534E+02	0.4830449879E-01	0.4474348402E-01	0.4474348402E-01
1	5	0.5277489255E+01	-0.4389237657E+02	0.4486782793E-01	0.4486782793E-01	0.4486782793E-01
1	6	0.6594794555E+01	-0.4396965451E+02	0.8881334200E-01	0.8881334200E-01	0.8881334200E-01
1	7	0.791199993E+01	-0.4404244254E+02	0.5849229655E-01	0.5849229655E-01	0.5849229655E-01
1	8	0.9229437211E+01	-0.4405831588E+02	0.2414216682E-01	0.2414216682E-01	0.0000000000E+00
2	1	0.0000000000E+00	-0.4399458479E+02	0.1112676571E-01	0.1112676571E-01	0.0000000000E+00
2	2	0.1356841287E+01	-0.4398451885E+02	0.4270600251E-01	0.4018848625E-01	0.4018734489E-01
2	3	0.2677587331E+01	-0.4394948430E+02	0.5479419750E-01	0.5096369018E-01	0.5096445426E-01
2	4	0.4004269114E+01	-0.4390739111E+02	0.5004508819E-01	0.4463448973E-01	0.4464878761E-01
2	5	0.5328036512E+01	-0.4389409127E+02	0.4291037894E-01	0.4291037894E-01	0.4291037894E-01
2	6	0.6642907129E+01	-0.4397034020E+02	0.8879366098E-01	0.8879366098E-01	0.8879366098E-01

2	7	0.7959713712E+01	-0.4404290631E+02	0.5713917408E-01	0.5713917408E-01	0.5713917408E-01
2	8	0.9278358213E+01	-0.4405831588E+02	0.2414216682E-01	0.2414216682E-01	0.0000000000E+00
3	1	0.0000000000E+00	-0.4399458479E+02	0.1112676571E-01	0.1112676571E-01	0.0000000000E+00
3	2	0.1356624500E+01	-0.4399408010E+02	0.1114085905E-01	0.1114085905E-01	0.1114085905E-01
3	3	0.2730952540E+01	-0.4397302719E+02	0.5096325231E-01	0.4680553493E-01	0.4683808222E-01
3	4	0.4090362450E+01	-0.4392669466E+02	0.5272530274E-01	0.4351975945E-01	0.4355359239E-01
3	5	0.5418808773E+01	-0.4389735067E+02	0.3886543373E-01	0.3886543373E-01	0.3886543373E-01
3	6	0.6726370673E+01	-0.4397144829E+02	0.8809362538E-01	0.8809362538E-01	0.8809362538E-01
3	7	0.8041492838E+01	-0.4404354368E+02	0.5543086596E-01	0.5543086596E-01	0.5543086596E-01
.....						
.....						

各行に 1 つのレプリカに関するエネルギーや力の情報が出力されます。1 列目が NEB ステップ数, 2 列目がレプリカの ID, 3 列目が 0 番目のレプリカからの“距離”, 4 列目がレプリカのエネルギー, 5 列目がレプリカに働く力の最大値, 6 列目が NEB 力の最大値, 7 列目がレプリカに働く力の最大値を経路に射影した力 (NEB 力の計算に利用される力) の最大値に対応します。

- nfnebdynm.data

座標データの履歴が記録されます。通常の PHASE の計算で得られる nfdynm.data ファイルと比較すると簡略化された形式で出力されます。具体的には以下のような形式で出力されます。

#step	image	atom	cps			
0	1	1	0.0000000000	0.0000000000	0.0000000000	0.0000000000
0	1	2	5.2000000098	5.2000000098	0.0000000000	0.0000000000
0	1	3	2.6000000049	2.6000000049	2.6000000049	2.6000000049
0	1	4	7.8000000147	7.8000000147	2.6000000049	2.6000000049
0	1	5	0.0000000000	5.2000000098	5.2000000098	5.2000000098
0	1	6	5.2000000098	0.0000000000	5.2000000098	5.2000000098
0	1	7	2.6000000049	7.8000000147	7.8000000147	7.8000000147
0	1	8	7.8000000147	2.6000000049	7.8000000147	7.8000000147
0	1	9	0.0000000000	0.0000000000	10.4000000197	10.4000000197
0	1	10	5.2000000098	5.2000000098	10.4000000197	10.4000000197
0	1	11	2.6000000049	2.6000000049	13.0000000246	13.0000000246
0	1	12	3.1080442326	2.0700339938	20.7869859136	20.7869859136
0	1	13	2.0700340645	3.1080442772	20.7869859136	20.7869859136
0	2	1	0.0000000000	0.0000000000	0.0000000000	0.0000000000
0	2	2	5.2000000098	5.2000000098	0.0000000000	0.0000000000
0	2	3	2.6000000049	2.6000000049	2.6000000049	2.6000000049
0	2	4	7.8000000147	7.8000000147	2.6000000049	2.6000000049
0	2	5	0.0000000000	5.2000000098	5.2000000098	5.2000000098
0	2	6	5.2000000098	0.0000000000	5.2000000098	5.2000000098
0	2	7	2.6000000049	7.8000000147	7.8000000147	7.8000000147
0	2	8	7.8000000147	2.6000000049	7.8000000147	7.8000000147
0	2	9	0.0000000000	0.0000000000	10.4000000197	10.4000000197
0	2	10	5.2000000098	5.2000000098	10.4000000197	10.4000000197
0	2	11	2.6000000049	2.6000000049	13.0000000246	13.0000000246
0	2	12	3.2652054480	1.9060914168	19.8836995566	19.8836995566
0	2	13	1.9060915098	3.2652055024	19.8836994729	19.8836994729

各行が, ある NEB ステップ・あるレプリカ・ある原子の座標データに対応します。1 列目が NEB ステップ, 2 列目がレプリカの ID, 3 列目がレプリカ内における原子の ID, 4, 5, 6 列目が原子座標です。座標は, ボーア単位, カルテシアン座標で出力されます。

nfefn.data ファイルと nfdynm.data ファイルは通常の PHASE の計算においてはそれぞれエネルギーおよび座標値の履歴が記録されるファイルですが, NEB 計算の場合は最新のレプリカ列のエネルギーおよび座標データが記録されたファイルです。nfefn.data ファイルには nfnebenf に記録されたデータの最後の NEB ステップのデータが記録されます。nfdynm.data ファイルは, PHASE の通常の形式で記録されますが, 通常の計算では構造最適化や分子動力学シミュレーションの履歴となるところがレプリカ列になります。

5.4.1.5 計算例：シリコン表面に水素分子が解離吸着する反応

シリコン表面に水素分子が解離吸着する反応の例題の入力ファイルは、`samples/neb` 以下にあります。ここで紹介する例題は、シリコン表面に水素分子が解離吸着する反応をシミュレートします。始状態は表面と表面から十分離れた場所にある水素分子から成る系、始状態は表面のシリコン原子に水素分子が解離し、吸着した系です。始状態と終状態の構造をそれぞれ図 5-49、図 5-50 とに示します。ただし、あくまで例題ですので、通常は実行する始状態、終状態の構造最適化は実行していません。

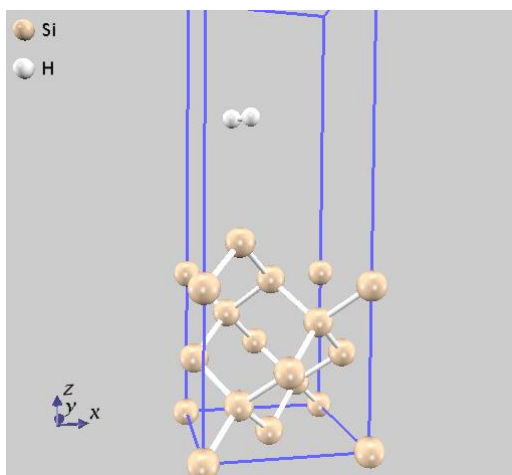


図 5-49 本例題の始状態

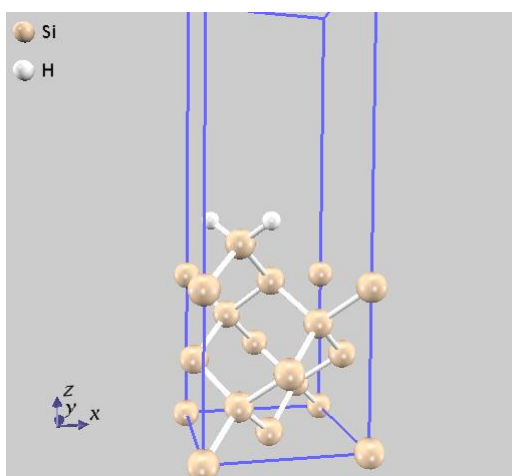


図 5-50 本例題の終状態

(1) 入力ファイル

`control` ブロックにおいて、全体的な計算条件の指定を行います。

```
Control{
  condition = initial ! {initial|continuation|automatic}
  cpumax = 1 day ! {sec|min|hour|day}
  max_iteration = 10000000
  multiple_replica_mode = ON
  multiple_replica_max_iteration = 2000
}
```

`multiple_replica_mode` に `ON` を指定することにより、NEB の計算が実行されます。また、NEB の繰り返

し計算の上限回数を `multiple_replica_max_iteration` 変数によって **2000** としています。
`multiple_replica` ブロックの下に `structure` ブロックにおいてレプリカの指定を実行しています。以下のようになります。

```
multiple_replica{
  ....
  structure{
    number_of_replicas = 6
    replicas{
      #tag replica_number howtogive_coordinates end0 endl
      1          proportional          0      -1 ! 0: end0, -1:endl
      2          proportional          0      -1
      3          proportional          0      -1
      4          proportional          0      -1
      5          proportional          0      -1
      6          proportional          0      -1
    }
    endpoint_images = directin ! {no or nothing | file | directin}
    howtogive_coordinates = from_endpoint_images
    atom_list_end0{
      coordinate_system = cartesian ! {internal|cartesian}
      atoms{
        #units angstrom
        #tag element rx ry rz
        Si  0.000000000000      0.000000000000      0.000000000000
        ...
        ...
      }
    }
    atom_list_end1{
      coordinate_system = cartesian ! {internal|cartesian}
      atoms{
        #units angstrom
        #tag element rx ry rz
        Si  0.000000000000      0.000000000000      0.000000000000
        ...
        ...
      }
    }
  }
  ....
}
```

`number_of_replicas` に **6** と指定していますが、この指定によってレプリカ数を合計 **6** としています。
`replicas` ブロックにおいて実際にどのようにレプリカの座標を作るかを指定しています。この例では、すべて始状態・終状態の線形補完によって作る、という設定になります。 `atom_list_end0` および `atom_list_end1` ブロックには始状態・終状態の座標値を指定しています。この指定は、前記の通り通常の **PHASE** の座標指定と変わるところはありません。

```
multiple_replica{
  ...
  accuracy{
    dt = 40 au_time
    neb_time_integral = quench
    penalty_function = off
    neb_convergence_condition = 3
    neb_convergence_threshold = 5.0e-04
  }
}
```

(2) 計算結果

本例題を実行すると得られる結果を紹介します。

図 5-51 に、本例題を実行すると得られる、NEB の繰り返し計算と NEB 力の最大値の関係を示します。はじめのうちは大きな力が働いていますが、計算が進行するにつれて小さくなっていき、41 回の繰り返し計算の後収束判定を満たして計算が終了しています。

図 5-52 に、本例題を実行すると得られる各イメージとエネルギーの関係を示します。この図より、遷移状態は 4 番目のレプリカであり、始状態から見ると障壁エネルギーが約 1.08 eV であることが分かります。

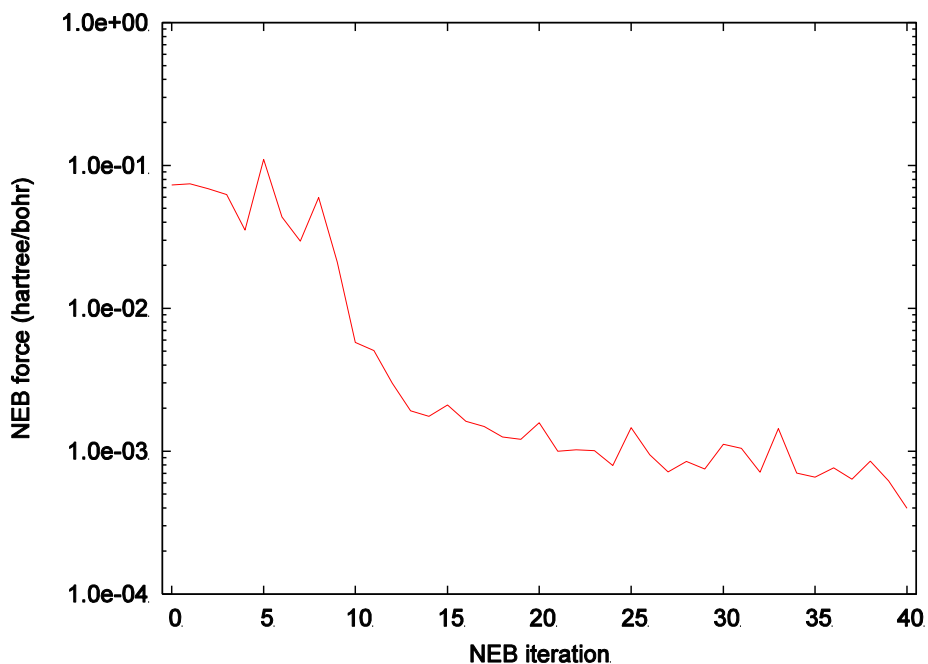


図 5-51 NEB 力の履歴

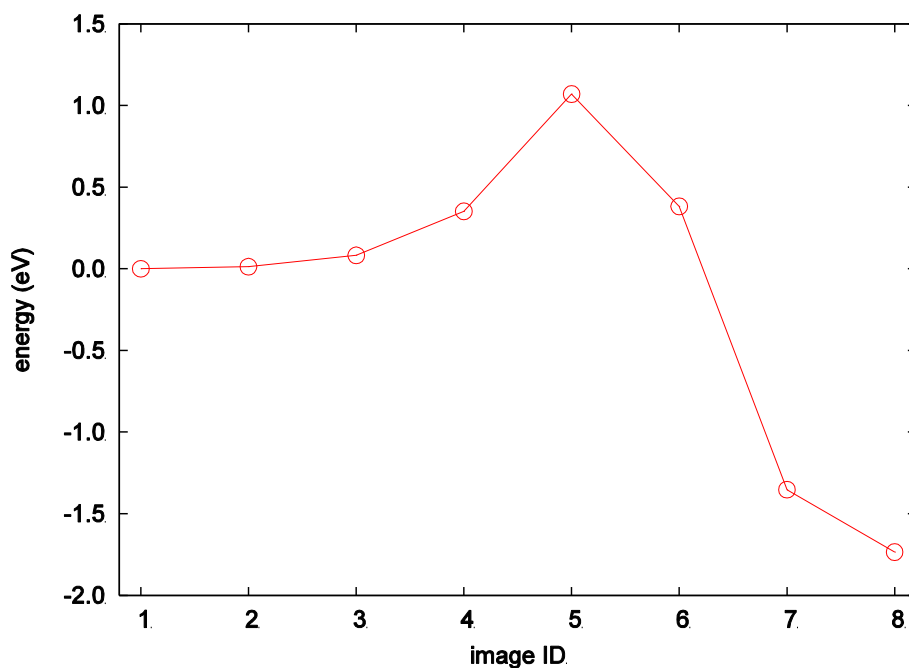


図 5-52 最終的に得られる反応経路と各レプリカのエネルギーの関係

図 5-53 に、遷移状態における原子配置を示します。この図から明らかなように、本例題では「水素分子が解離、そして吸着する」直前の構造が 遷移状態です。

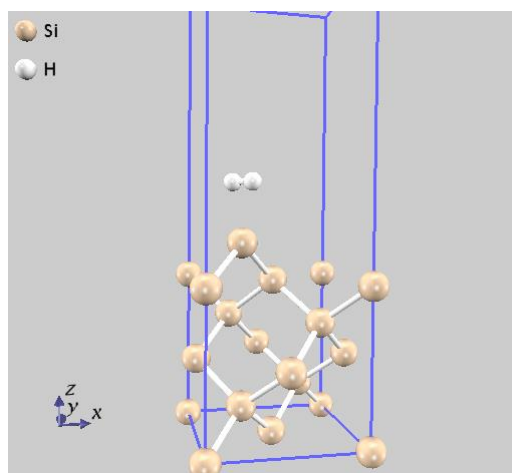


図 5-53 遷移状態における原子配置

5.4.1.6 使用における注意点

- レプリカ並列

NEB 法は「レプリカ並列」に対応しています。レプリカ並列機能を使用するためには、引数に通常の `ne=NE nk=NK` に加え、並列したいレプリカ数を `NR` とすると `nr=NR` を加えます。MPI プロセス数は `NE x NK x NR` と等しい必要があります。たとえば、以下のようなコマンドになります。

```
% mpirun -n N phase ne=NE nk=NK nr=NR
```

- 計算の停止と継続計算

NEB 法は計算の停止と継続計算に対応していますが、通常の計算とは異なる手続きが必要です。

- 計算のストップ

入力データの `multi_replica_max_iteration`, または `nfnebstop.data` に記述された NEB の `iteration` 数で NEB 計算は終了します。また、各イメージの電子状態計算において、入力データの `max_iteration`, `cpumax`, `nfstop.data` ファイルの設定によっても計算は終了します。いずれの場合でも、停止した箇所からリスタートすることが可能です。

計算ストップ時における通常の PHASE との相違点を挙げます。PHASE では、`nfstop.data` ファイルによって終了した場合、`nfstop.data` ファイルは空ファイルとなります。他方 NEB 計算では、あるイメージを `nfstop.data` によって終了した場合、`nfstop.data` はただちには空ファイルとはならず、ほかのイメージの計算を行います。NEB 計算終了処理においてはじめて `nfstop.data` ファイルを空ファイルとします。

- 計算のリスタート

PHASE と同様、入力データにおいて、`condition` の値を `continuation` とすることによってリスタート計算を行います。

```
Control{
    condition = continuation
    ...
    ...
}
```

リスタート時に利用するファイルは下記のファイルです。

- NEB計算: `neb_continue.data`
- 電子状態計算: 各レプリカのPHASE用リスタートファイル;

`neb_continue.data, continue.data_r*, continue_bin.data_r*, zaj.data_r*, nfchgt.data_r*`

5.4.1.7 参考文献

- [1] G. Mills and H. Jónsson, "Quantum and Thermal Effects in H₂ Dissociative Adsorption: Evaluation of Free Energy Barriers in Multidimensional Quantum Systems" *Phys. Rev. Lett.* **72** (1994) p. 1124.
- [2] G. Henkelman, B. P. Uberuaga and H. Jónsson, "A climbing image nudged elastic band method for finding saddle points and minimum energy paths" *J. Chem. Phys.* **113** (2000) p. 9901.

5.4.2 拘束条件付きダイナミクスと Blue Moon 法による自由エネルギー解析

5.4.2.1 機能の概要

化学反応経路を探索する手法として、ボンド長やボンド角などの化学反応を特徴づける「反応座標」を導入し、想定した反応経路上でその値を逐次変化させながら 反応座標を拘束した構造最適化や分子動力学シミュレーションを実施する、という手法があります。単純な構造最適化の場合絶対零度における反応経路が得られ、有限温度の分子動力学シミュレーションを実施すると自由エネルギー差が得られます。ここでは、PHASE を利用して拘束条件付きダイナミクスを追跡する方法を説明します。

5.4.2.2 入力パラメータ

本機能と関連あるタグの一覧を表 14 に示します。

拘束条件付きダイナミクスに関連のあるタグの一覧

第1ブロック識別子	第2, 第3ブロック識別子	タグ識別子	説明
control			
		driver	ダイナミクスの種類を選択する変数。 拘束条件付きダイナミクスの場合 <code>constraints</code> を指定する。
structure			原子座標データの指定を行うブロック
	constrainablexx		拘束条件を定義するブロック。 xx には拘束条件を識別するための整数を 1 はじまりで指定
		type	拘束条件の“種類”の指定。 bond_length, bond_angle, dihedral_angle bond_length_diff, bond_angle_diff, distance_from_pos, plane, center_of_mass, coordination_number
		atomx	拘束条件が関わる原子を指定する。x は整数で、たとえば type = bond_length の場合 2 原子が拘束に関わるので atom1 と atom2 で指定する。
		mobile	拘束条件が“可動化否か”を指定する。on とすると可動、 off とすると不動。デフォルト値は off
		monitor	指定の拘束条件を“監視”するかどうかを指定する真偽値。 デフォルト値は off
	reaction_coordinate		指定の拘束条件が“反応座標”としたい場合に作成するブ ロック。
		sw_reaction_coordinate	on の場合反応座標とみなされる。
		init_value	反応経路の初期値に対応する単位で指定。
		final_value	反応経路の最終値に対応する単位で指定。
		increment	final_value, init_value の刻み幅を指定。
	plane		面内拘束における面の法線ベクトルを指定するブロック
		normx, normy, normz	法線ベクトルの x, y, z 成分
	distance_from_pos		場所の指定を行うブロック
		posx, posy, posz	指定したい場所の x, y, z 成分
	coordination_number		配位数指定を行うブロック
		kappa_inv	配位数定義式の K の逆数を長さの単位で指定
		kappa	配位数定義式の K を 1/bohr 単位で指定
		rcut	配位数定義式の r_c の値を長さの単位で指定
	center_of_mass		重心を変化させる方向を指定するブロック
		directionx	指定したい方向の x 成分
		directiony	指定したい方向の y 成分
		directionz	指定したい方向の z 成分
structure_evolution			原子ダイナミクスの設定を行うブロック
		method	原子ダイナミクスの方法を指定する。 拘束条件付きダイナミクスの場合 quench, damp, velocity_verlet, temperature_contro のみ利用可能。

拘束条件付きダイナミクスを実行するには、まず以下の要領で **condition** ブロックの下で **driver** 変数を指定します。

```
condition{
  ...
  driver=constraints
  ...
}
```

次に、以下のように **structure** ブロックの下に **constrainablexx** ブロックを作成します。ここで **xx** は整数です。

```
structure{
  ...
  ...
  constrainable1{
    type=bond_length
    atom1=1
    atom2=2
    mobile = off
    monitor = off
    reaction_coordinate{
      sw_reaction_coordinate=on
      init_value = 2.4 angstrom
      increment  = 0.1 angstrom
      final_value = 8.0 angstrom
    }
    plane{
      normx=1
      normy=0
      normz=0
    }
    coordination_number{
      kappa = 5.0
      rc    = 2.0 angstrom
    }
  }
  ...
  ...
}
```

拘束条件はいくつでも指定可能ですが、たとえば **constrainable1**, **constrainable2**, **constrainable4** という 3 つの **constrainablexx** ブロックが存在する場合、**constrainable4** ブロックは入力解釈の対象にはなりません。また、互いに相いれない拘束条件を定義してしまうと、拘束条件を課すための収束計算が破たんしてしまう場合がありますので注意が必要です。**constrainablexx** ブロックの下では、以下の変数/ブロックを指定することができます。

type 変数

bond_length
bond_angle
dihedral_angle
bond_length_diff
bond_angle_diff
distance_from_pos
plane
center_of_mass
coordination_number

拘束条件の“種類”を指定します。以下のいずれかの値をとります。

- 2 原子間の距離を拘束します。
- 3 原子の成すボンド角を拘束します。
- 4 原子の 2 面角を拘束します。
- 2 原子間の距離の差を拘束します。
- 3 原子が成す角度の差を拘束します。
- 指定の場所からの距離を拘束します。
- 面内に指定の原子を拘束します。
- 指定の原子群の重心を拘束します。
- 配位数を拘束します。ここで、0 番目の原子の周りの「配位数」は以下の式によって評価します。

$$\sigma = \sum_{i \neq 0} S(|\mathbf{r}_i - \mathbf{r}_0|),$$

$$\sigma = \sum_{i \neq 0} S(|\mathbf{r}_i - \mathbf{r}_0|),$$

κ , r_c はパラメーターであり, first coordination shell でほどよく 0 に近づくように決定します。

atomx 変数	指定の拘束条件に関わる原子を指定します。x は数字であり, たとえば 2 原子間の距離の場合は 2 つの原子が拘束に関わるので, atom1 と atom2 に対応する原子の番号を指定します。type が coordination_number の場合, 配位数を計算する中心の原子の番号を指定します。
mobile 変数	指定の拘束条件が“可動か否か”を指定するためのスイッチです。off とすると拘束され, on とすると拘束はされません。デフォルト値は off です。
monitor 変数	指定の拘束条件を“監視” (値を算出し, ログファイルに出力) するか否かを指定するスイッチです。デフォルト値は off です。
reaction_coordinate ブロック	指定の拘束条件が, “反応座標(逐次値を変化させられる)”である場合に作成するブロックです。以下の変数を指定することができます。
sw_reaction_coordinate init_value	on の際, 反応座標とみなされます。 反応経路の初期値を, 対応する単位で指定します。指定がない場合, 入力原子配置から求められる値が採用されます。この値と, 入力原子配置から求められる値が異なる場合, 入力原子配置が修正されたのちに計算が実行されます。このため, 一回目の(拘束力も含む)原子に働く力の最大値が見掛け上非常に大きな値となることがありますが, これは正常な振る舞いです。
final_value increment	反応経路の最終値を対応する単位で指定します。 final_value と init_value の間の刻み幅を指定します。 反応座標を逐次変化させる場合, 以下のケースは特殊であるので注意が必要です。
type が plane の場合	この場合, 変化するのは面の原点です。原点の座標値は, 指定の法線ベクトルと原子の座標値からプログラムが自動的に決めますが, 反応座標の変化としては, 法線ベクトルの方向に原点が変化する, という振る舞いになります。 init_value, final_value, increment は, この原点の移動量を指定してください。なお, この場合の init_value のデフォルト値は 0 ですが, 通常明示的に指定する必要はないはずです。
type が center_of_mass の場合	この場合, 重心を, 指定の方向に移動させます。init_value, final_value, increment は, この移動量を指定してください。なお, この場合の init_value のデフォルト値は 0 ですが, 通常明示的に指定する必要はないはずです。
plane ブロック	
normx	法線ベクトルの x 座標。
normy	法線ベクトルの y 座標。
normz	法線ベクトルの z 座標。
distance_from_pos ブロック	type として distance_from_pos を採用する場合の, 場所の指定を行うブロックです。次の変数を指定することができます。
posx	指定したい場所の, x 座標を長さの単位で指定します。

	posy	指定したい場所の、y座標を長さの単位で指定します。
	posz	指定したい場所の、z座標を長さの単位で指定します。
coordination_number ブロック		配位数拘束の定義式における κ, r_c の値を指定するブロックです。次の変数を指定することができます。
	kappa_inv	κ の値を、長さの単位で指定します。
	kappa	κ の値をの値を、1/bohr 単位で指定します。kappa_inv よりも優先されます。
	rcut	r_c の値を、長さの単位で指定します。
center_of_mass ブロ ック		type として center_of_mass を採用し、かつ反応座標を変化させる場合、「変化させる方向」をここで指定します。
	directionx	変化させる方向のx座標を指定します。
	directiony	変化させる方向のy座標を指定します。
	directionz	変化させる方向のz座標を指定します。

拘束条件の指定の次は、採用するダイナミクスのアルゴリズムを指定します。通常の **PHASE** の入力と同様、**structure_evolution** ブロックの下で行います。

```
structure_evolution{
  method=quench
  dt=40
  ...
}
```

ここで、method としては quench, damp, velocity_verlet, temperature_control を利用することができます。拘束条件を課している場合、gdiis, cg などには現バージョンでは未対応なのでご注意ください。また、damp は damped molecular dynamics 法による構造最適化を実施する場合に指定します。この手法は、多くの場合単純な quenched MD よりは大きな時間刻み(dt)を採用することができ、速く収束させることのできる手法です。

次に、反応座標の変化の指定方法について説明します。

- 複数の反応座標を逐次変化させる方法

複数の反応座標を逐次変化させる場合のプログラムの振る舞いを説明します。たとえば以下のように入力で指定した場合について説明します。

```
structure{
  ....
  ....
  constrainable1{
    mobile = off
    monitor = on
    type = dihedral_angle
    atom1 = 2
    atom2 = 4
    atom3 = 3
    atom4 = 1
    reaction_coordinate{
      sw_reaction_coordinate = on
      init_value = -179 degree
      final_value = -1 degree
      increment = 5 degree
    }
  }
  constrainable2{
    type=bond_length
    monitor=on
    atom1=3
```

```

    atom2=4
    reaction_coordinate{
      sw_reaction_coordinate=on
      init_value = 1.2 angstrom
      final_value = 1.6 angstrom
      increment = 0.05 angstrom
    }
  }
  ....
  ....
}

```

まず, constrainable1 ブロックにおいて 2 面角を -179° から -1° まで 5° 刻みで変化させるように指定しています。さらに, constrainable2 ブロックにおいてはボンド長を 1.2 \AA から 1.6 \AA まで 0.05 \AA 刻みで変化させるように指定をしています。このような入力を記述した場合, まずボンド長を 1.2 \AA に固定した状態で 2 面角を -179° から -1° まで変化させて計算が実行されます。 -1° の計算が終了したら, 次はボンド長を 1.25 \AA に変化させ, 今度は -1° から -179° まで 2 面角を変化させる計算を実行します。このような変化のさせ方を採用することによって, 隣り合う反応座標の組の間で原子配置が 極端に変化することを防いでいます。

以上のような方針で検討する反応座標が決まりますので, 合計すると検討する反応座標の数は反応座標 α において検討する反応座標の数を n_α とすると $\prod_\alpha n_\alpha$ となります。これよりもきめ細やかに反応座標の組を指定するには, 次に説明する「反応座標の変化の仕方をファイルを介して指定する」機能を利用します。

● ファイルから反応座標の変化の仕方を指定する方法

拘束条件の変化のさせ方は上述の reaction_coordinate ブロックにおいて指定しますが, この方法の場合は等間隔の指定です。特に前述の, 複数の反応座標を変化させる計算においては, 反応座標 α において検討する反応座標の数を n_α とすると $\prod_\alpha n_\alpha$ 個の反応座標を検討することになり, 計算時間が膨大になることがあります。このような制限が問題となる場合に, 「反応座標 (の組)」をファイルから指定することが可能となっています。

まず, 検討したい拘束条件を constrainablexx ブロックにおいて通常通り指定します。次に, structure ブロックの下に以下の変数を定義します。

```

structure{
  ....
  reac_coord_generation = via_file
  ....
}

```

最後に, 作業ディレクトリーに reac_coords.data というファイル名のファイルを作成し, 次のような内容を記述します。

1	-1.9373154697	2.2676711906
2	-1.7627825445	2.2676711906
3	-1.5882496193	2.2676711906
4	-1.4137166941	2.2676711906
5	-1.2391837689	2.2676711906
6	-1.0646508437	2.2676711906
7	-0.8901179185	2.2676711906
8	-0.7155849933	2.2676711906
9	-0.7155849933	2.3621574902
10	-0.8901179185	2.3621574902
11	-1.0646508437	2.3621574902
12	-1.2391837689	2.3621574902
13	-1.4137166941	2.3621574902
14	-1.5882496193	2.3621574902
15	-1.7627825445	2.3621574902
16	-1.9373154697	2.3621574902
17	-1.9373154697	2.4566437898
18	-1.7627825445	2.4566437898

19	-1.5882496193	2.4566437898
20	-1.4137166941	2.4566437898
21	-1.2391837689	2.4566437898
22	-1.0646508437	2.4566437898
23	-0.8901179185	2.4566437898
24	-0.7155849933	2.4566437898
	
	
	

各行が 1 つの反応座標の組に相当します。行の 1 列目にはその反応座標の組を識別するための番号を整数で入力します。2 列目以降は、定義した拘束条件の順番で反応座標の値を入力します。この例では 2 種類の反応座標を検討していることになります。1 つ目の反応座標の組では 1 番目の拘束条件として“-1.9373154697”という値、2 番目の拘束条件として“2.26711906”という値を指定することになります。単位は、PHASE のデフォルトの単位を利用して指定するようにしてください。長さならば bohr 単位、角度なら radian 単位です。

5.4.2.3 計算の実行方法

拘束条件付きダイナミクスは、「反応座標を逐次変化させて計算する」ケースにおいては原子配置に対する並列計算に対応しています。PHASE を以下のように起動してください。

```
% mpirun -np NP phase ne=NE nk=NK nr=NR
```

ここで、NP が MPI プロセス数、NE がバンド並列数、NK が k 点並列数、NR が原子配置並列数であり、 $NP = NE \times NK \times NR$ という関係が成立している必要があります。この機能を利用する場合、継続計算の処理がプログラム内で若干変化するので、継続計算間で nr を指定したりしなかったりするとエラーとなる点にご注意ください。nr は 1 でも構わないので、原子配置並列を継続計算のあるタイミングで無効にする場合、NR を 1 とすれば目的の動作を達成することができます。

5.4.2.4 計算結果の出力

出力ファイルは、「反応座標を逐次変化させる」機能を利用していない場合は通常の PHASE の出力と同様です。すなわち、file_names.data ファイルにおいて F_ENF 識別子によって指定されるファイルに各ステップにおけるエネルギーや原子に働く力の最大値が、F_DYNM 識別子によって指定されるファイルに各ステップにおける原子配置や各原子に働く力が出力されます。ただし、「原子に働く力の最大値」は、拘束条件を課すために必要な「拘束力」も含む点に注意が必要です。

他方、「反応座標を逐次変化させる」計算を実行している場合、次のようなファイル群が出力されます(ここで、F_ENF 識別子によって指定されるファイルのファイル名を nfefn.data、F_DYNM 識別子によって指定されるファイルのファイル名を nfdynm.data とします)。

nfefn.data.reacxx	xx 番目の反応座標の、各ステップにおけるエネルギーおよび原子に働く力の最大値のデータが出力されます。
nfefn.data.converged (構造最適化の場合のみ)	各反応座標において、反応座標の値そのものと、収束したエネルギーおよび原子に働く力の最大値が出力されます。反応座標とエネルギーの関係をプロットすることによって、「反応経路」と「エネルギー」の関係を解析することができます。
nfdynm.data.reacxx	xx 番目の反応座標の、各ステップにおける原子配置や原子に働く力が出力されます。
nfdynm.data.converged (構造最適化の場合のみ)	各反応座標において、構造最適化が収束した後の原子配置が出力されます。
nfbluemoon.data.reacxx (分子動力学シミュレーションの場合のみ)	xx 番目の反応座標の、自由エネルギー差を算出するために必要なラグランジュの未定乗数の値が記録されます。

さらに、継続計算ファイルや波動関数・電荷密度ファイルなどは反応座標ごとに出力されます。

5.4.2.5 Blue Moon 法による自由エネルギーの計算

(1) 機能の概要

拘束条件付きの分子動力学シミュレーションが発生する統計集合(blue moon ensemble)のデータを利用すると、検討した反応座標の経路上における自由エネルギーの変化を算出することができます[1]。反応座標が ξ_1 から ξ_2 へ変化する場合の自由エネルギー差は、次のように計算することが可能です。

$$W(\xi_1) - W(\xi_2) = \int_{\xi_2}^{\xi_1} d\xi \frac{\partial W}{\partial \xi}.$$

ここで自由エネルギーの反応座標微分、 $\left\langle \frac{\partial W}{\partial \xi} \right\rangle_{\xi}$ は mean force と呼ばれる物理量であり、ハミルトニアン H の反応座標微分と次のような関係があります。

$$\frac{\partial W}{\partial \xi} = \left\langle \frac{\partial H}{\partial \xi} \right\rangle_{\xi}^{\text{cond}}.$$

ここで $\langle \dots \rangle^{\text{cond}}$ とは「条件付き統計平均」です。拘束条件付き分子動力学シミュレーションの統計平均と条件付き統計平均は単純には結びつきませんが、(21)式は拘束条件付き分子動力学を遂行する際に計算するラグランジュの未定乗数 λ を利用して以下のように計算することができます。

$$\begin{aligned} \frac{\partial W}{\partial \xi} &= - \frac{\langle |\mathcal{E}|^{-1/2} \lambda \rangle}{\langle |\mathcal{E}|^{-1/2} \rangle} \\ \Xi &= \sum_i \frac{1}{m_i} \frac{\partial \xi}{\partial \vec{r}_i} \frac{\partial \xi}{\partial \vec{r}_i} \end{aligned}$$

(22)式には、厳密にはより複雑な補正項がつきますが、実用上は問題ないとされています。

PHASEによる拘束条件付き分子動力学シミュレーションの結果から自由エネルギー差を計算するには、PHASEパッケージに付属している **bluemoon** プログラムを利用します。

現バージョンでは、**bluemoon** プログラムは反応座標が1つの場合のみに対応しています。

(2) bluemoon プログラムのコンパイル

bluemoon プログラムのソースコードは、PHASE インストールディレクトリーの **src_bm** ディレクトリーに納められています。**bluemoon** プログラムは Fortran90 コンパイラーと C コンパイラーを必要とします。Fortran90 コンパイラーを環境変数 **F90** に、C コンパイラーを環境変数 **CC** に設定し、**make** コマンドを発行すればコンパイルすることができます。以下はお使いのシステムが **bash** で、Fortran コンパイラーのコマンドが **f90**、C コンパイラーのコマンドが **cc** の場合の例です。

```
% cd phase0_2019.01
% cd src_bm
% export F90=f90
% export CC=cc
% make
% make install
```

環境変数 **F90** と **CC** の指定がない場合、**gfortran** と **gcc** がデフォルト値として利用されます。コンパイルが終了すると、**bluemoon** という名前の小さなプログラムが作成されます。 **% make install** とすると **phase0_2019.01/bin** ディレクトリーの下に **bluemoon** を移すことができます。

(3) blumoon プログラムの入力パラメータ

blumoon プログラムの入力ファイルは、PHASE のそれと同等です。nfnp.data ファイルに thermodynamic_integration ブロックを作成し、計算条件を入力します。たとえば以下ようになります。

```
thermodynamic_integration{
  nsteps=2000
  nequib=1000
  istart_reac_coords=1
  nreac_coords=14
  nsample=10
  smooth=off
  basedir=.
}
```

thermodynamic_integration ブロックでは以下の指定を行うことができます。

nsteps	各反応座標における分子動力学シミュレーションの総ステップ数を指定します。デフォルト値は 2000 ですが、実施した計算に合わせて変更してください。
nequib	nsteps の内、平衡化のため捨てるステップ数を指定します。nsteps よりも小さく、熱平衡に至ったと考えられる値を指定してください。
istart_reac_coords	最初に検討する反応座標の ID を入力します。デフォルト値は 1 です。
nreac_coords	最後に検討する反応座標の数を指定します。
nsample	統計誤差を見積もる場合にシミュレーションを何分割するかを指定します。
smooth	on とすると、三次のスプライン関数によって計算結果を滑らかにします。
basedir	結果を出力するディレクトリーを指定します。デフォルト値はカレントディレクトリーです。

(4) blumoon プログラムの実行方法

以上のような入力を作成したら、次のように blumoon を走らせます。

```
% blumoon inputfile
```

引数で指定する inputfile は入力ファイルのファイル名です。指定がない場合、nfnp.data という文字列が採用されます。

(5) 計算結果の出力

計算が終了すると、次のファイルが作成されます。

● potential_of_mean_force.data

自由エネルギーの計算結果が出力されます。以下のような形式で出力されます。

```
#value, potential of mean force in Hartree, eV, kcal/mol, kJ/mol
2.4566437898 -0.0215821952 0.0003443042 -0.5872816633 0.0093689992 -13.5430301648
0.2160541460 -56.6640534911 0.9039707906
2.2676711910 -0.0224669448 0.0003796767 -0.6113569350 0.0103315334 -14.0982188431
0.2382507016 -58.9869635475 0.9968412043
2.0786985910 -0.0226882285 0.0004435350 -0.6173783747 0.0120692073 -14.2370764737
0.2783223931 -59.5679440305 1.1645012069
.....
.....
.....
```

各行が 1 つの反応座標のデータに相当します。1 列目が反応座標の値、2 列目、3 列目がハートリー単位、4 列目、5 列目が電子ボルト単位、6 行目と 7 行目が kcal/mol 単位、8 行目と 9 行目が kJ/mol 単位での自由エネルギーとその統計誤差の結果に対応します。

- ean_force_raw.data

検討した反応座標から得られる mean force の計算結果が出力されます。次のような形式で出力されます。

2.4566437898	0.0066082098	0.0188118786
2.2676711910	0.0034758686	0.0099291734
2.0786985910	-0.0009537509	0.0028573953
1.8897259920	-0.0074922663	0.0213420952
1.7007533930	-0.0098143395	0.0279585555
1.5117807940	-0.0157974842	0.0449758051
1.3228081950	-0.0161451965	0.0459534340
	
	
	

potential_of_mean_force.data ファイルと同様に、各行が 1 つの反応座標のデータに相当します。1 列目が反応座標の値、2 列目が mean force の値(単位 : hartree/対応する反応座標の単位)、3 列目が統計誤差に相当します。

- mean_force_smoothed.data

三次のスプライン関数によって自由エネルギー計算を滑らかにする場合 mean force を滑らかにしたあとに(20)式の積分を実施しますが、その滑らかにした mean force の計算結果が出力されます。そのデータ形式は、mean_force_raw.data ファイルから統計誤差の列を除いたものになります。

5.4.2.6 計算例 : H₂O₂ および H₂S₂ 分子の回転障壁の解析

拘束条件付き構造最適化計算の例として、H₂O₂ および H₂S₂ 分子の回転障壁の解析例を紹介します。H₂O₂, H₂S₂ は図 5-54 で示す分子構造を有する単純な分子です。HOOH (HSSH) が成す 2 面角の回転ポテンシャルは、H 原子同士の相互作用と H 原子と O(S) 原子の孤立原子対との相互作用が競合し、W 型ポテンシャルになることが知られています。2 面角を拘束した構造最適化を複数の 2 面角において実施することにより、このような振る舞いが得られるかどうかを確認します。

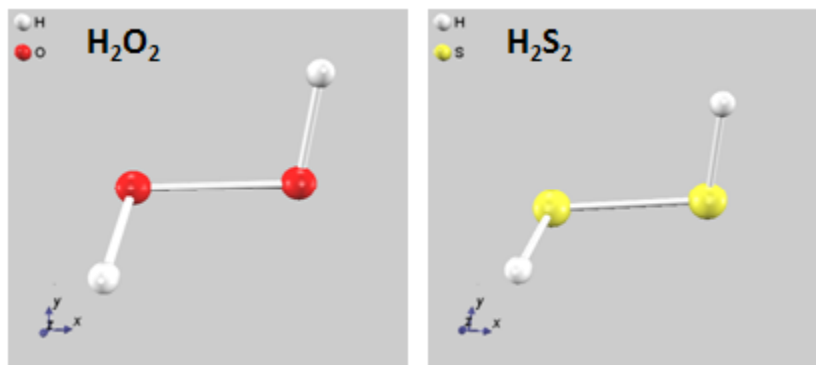


図 5-54 H₂O₂ および H₂S₂ 分子の分子構造

この例題の入力ファイルは、samples ディレクトリーの下に constraints 以下、H2O2 および H2S2 ディレクトリーにあります。まず、structure ブロックの下には以下の記述があります。

```
structure{
  constrainable1{
    type = dihedral_angle
    atom1 = 2
    atom2 = 4
    atom3 = 3
    atom4 = 1
    reaction_coordinate{
      sw_reaction_coordinate = on
      init_value = 9 degree
      final_value = 179 degree
      increment = 10 degree
    }
  }
}
```

```

    }
    ...
    ...
}

```

constrainable1 ブロックを作成し、その下で拘束条件の指定を行っています。この例題では拘束条件は一つのみ課しますが、互いに相いれる拘束条件であるならばいくつでも定義することが可能です。今は2面角の拘束を実施するので、**type** 変数としては **dihedral_angle** を指定しています。また、2面角を定義するために必要な4つの原子の番号を **atom1** から **atom4** 変数によって指定しています。さらに、**reaction_coordinate** ブロックを作成し、この拘束条件を逐次変化させる指定を行います。**sw_reaction_coordinate** を **on**, **init_value** と **final_value** をそれぞれ 9 degree と 179 degree, **increment** を 10 degree としていますが、このような指定によって、9°から179°まで、10°刻みで2面角を変化させて構造最適化を行います。

図 5-55 に、2面角と最適化の結果得られたエネルギーの関係を示します。図 5-55 には、実験結果[2]も併せて実線で表示しています。一見して明らかなように、計算結果と実験結果はよい一致が得られています（おおよそ 1 kcal/mol 程度の違い）。

H_2O_2 と H_2S_2 の大きな違いは2点あります。1点目は、安定な2面角の値です。 H_2O_2 は四面体の角度である 109.5° に近い値が安定であるのに対し、 H_2S_2 は 90° 付近が安定な2面角です。2点目は **trans** 障壁エネルギー（図 5-55 では 180° 付近の障壁エネルギー）の高さです。 H_2O_2 と比較すると、 H_2S_2 の **trans** 障壁ははるかに大きく、実験的には約6倍の値が得られています。いずれの点も本計算によって再現されており、妥当な結果が得られているものと考えられます。

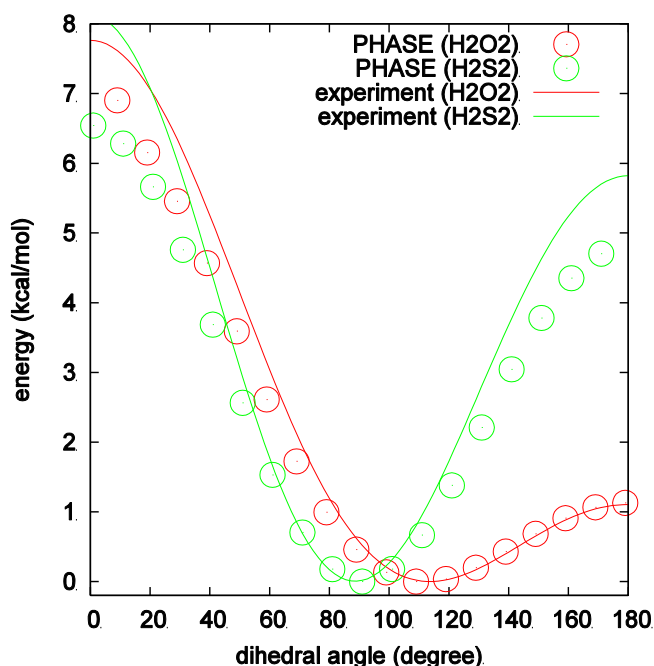


図 5-55 H_2O_2 および H_2S_2 分子の2面角とエネルギーの関係

5.4.2.7 使用における注意点

拘束条件付きダイナミクスは、全ての擬ポテンシャルと組み合わせて利用することができます。継続計算にも対応しています。また、反応座標に対する並列を実行することができます。反応座標に対して並列計算を行う場合、NEB法の場合と同様以下のようなコマンドを利用します。

```
% mpirun -n NP phase ne=NE nk=NK nr=NR
```

5.4.2.8 参考文献

- [1] Michiel Sprik and Giovanni Ciccotti, *Journal of Chemical Physics* **109** (1998) p. 7737.
- [2] G. Pelz, K. Yamada, and G. Winnewisser, *Journal of Molecular Spectroscopy* **159**, (1993) p. 507.

5.4.3 Meta-dynamics 法

5.4.3.1 機能の概要

Meta-dynamics 法[1], [2]は、化学反応などの障壁エネルギーの存在する過程を効率よく解析するための手法です。Meta-dynamics 法においては、 $S_{\alpha}(r)$ という“集団変数”を導入します。ここでいう集団変数とは、具体的には対象とする系の原子座標から定義可能な反応座標（ボンド長やボンド角などの内部座標や配位数など）を複数集めたものです。各集団変数には、仮想的な“粒子”が割り当てられます。この、“仮想的な粒子の運動”のことを Meta-dynamics とよびます。Meta-dynamics のアルゴリズムをうまく設計することによって、効率よく（検討している集団変数が作る）自由エネルギー表面を探索することができると考えられます。ここでは、PHASE に実装された Meta-dynamics 法の利用方法を説明します。

Meta-dynamics 法では、計算の履歴に依存するバイアスポテンシャル $V(t, s)$ をある間隔（通常数十から数百 MD ステップ）で足しこんでいきます。このような方針を採用することによって、自由エネルギー空間において一度訪れた点に訪れづらくする効果が発揮されます。十分長い時間シミュレーションを行うと $V(t, s)$ が自由エネルギー空間を埋め尽くしてしまい、反応は自由に起こることができるようになります。この状態に至る $V(t, s)$ （に-1を掛けた量）がすなわち自由エネルギーであるとみなすことができます。

Meta-dynamics 法によるシミュレーションの模式的な様子を図 5-56 に示します。この図では、まずシミュレーションは 1 の数字が割り当てられた谷から始まります。2 のバイアスを足し、さらに 3 のバイアスを足すと新しい局所極小（図中で最も左側の谷）に至ります。さらに 4, 5, 6 とバイアスポテンシャルを足すともっともエネルギーの低い谷（図中で最も右側の谷）へ至ることができます。最後に 7 のバイアスを足し、さらに 8 のバイアスポテンシャルまで足すと、系は集団変数の空間を自由に行き来できるようになります。この時点でのバイアスポテンシャルに-1を掛けると、それは自由エネルギーと見做すことができることが分かります。

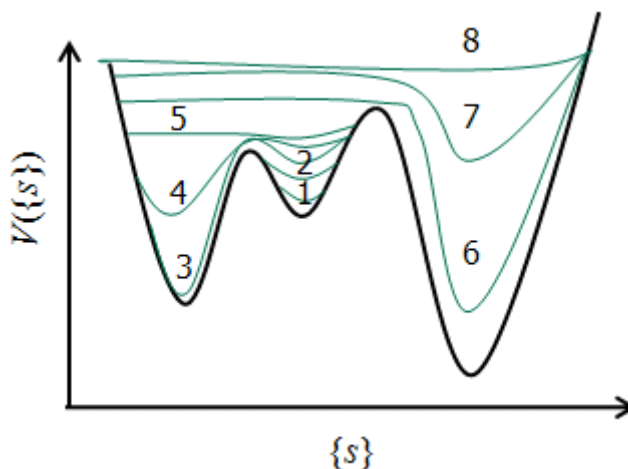


図 5-56 Meta-dynamics シミュレーションの模式図

Meta-dynamics 法の特徴として、反応座標と関連付けられた複数の動力学変数の動力学を追跡する点が挙げられます。この考え方を導入することによって、複数の反応座標を検討することが容易となり、また動力学変数自身が“もっともらしい”反応経路を探索してくれる効果が期待できます。また、blue moon 法の場合複数の反応座標を取り扱うことは（原理上は可能ではありますが）難しいのに対し、meta dynamics 法においては比較的容易です。したがって、反応座標が複数ある場合や、反応経路が自明でない場合などにおいて有効な方法であると考えられます。バイアスポテンシャル導入の方針の違いにより、自由エネルギー表面を粗く、すばやく探索することも、きめ細かく、精度よく探索することも可能です。Meta-dynamics 法シミュレーションは、あらかじめ決められた間隔でバイアスポテンシャルを足しながら進行していきます。この際にバイアスポテンシャルを構築するには、時刻 0 から現在までのデータをすべて利用して和を取る必要があるため、Meta-dynamics 法は $O(t^2)$ の計算手法となります（ただし、第一原理計算で利用するかぎりこの点が制約になることはないでしょう）。

Meta-dynamics 法のハミルトニアンは、具体的には、次のように記述されます。

$$H_{\text{meta}} = H_{\text{MD}} + \sum_{\alpha} \frac{1}{2} \mu_{\alpha} \dot{s}_{\alpha}^2 + \sum_{\alpha} \frac{1}{2} k_{\alpha} (S_{\alpha}(\mathbf{r}) - s_{\alpha})^2 + V(t, s),$$

$$V(t, s) = \sum_{t_i < t} w \exp \left[- \sum_{\alpha} \frac{(s_{\alpha}(t) - s(t_i))^2}{2 \Delta s_{\alpha}^2} \right]$$

ここで、 α は集団変数に含まれる各変数を識別する変数、 μ_{α} と s_{α} はそれぞれ仮想的な粒子の質量と座標、 $S_{\alpha}(\mathbf{r})$ は対象としているシステムから定義される「集団変数」、 k_{α} は仮想的な粒子の座標と集団変数を結びつける「ばね定数」、 $V(t, s)$ がバイアスポテンシャルです。足しこんでいったバイアスポテンシャルを記録しておく、そこから自由エネルギーを見積もることも可能です。このようなハミルトニアンから得られる動力学は、次のようにまとめることができます。

- 系は、集団変数を通して仮想的な粒子の座標値に緩く拘束される。
- 仮想的な粒子の座標は、バイアスポテンシャルの効果によって、すでに訪れた点には再訪づらい。

仮想的な粒子の座標の運動と系の運動の特徴的なタイムスケールが異なれば（仮想的な粒子の方が長いタイムスケールであれば）、系の運動は仮想的な粒子の運動の影響をそれほど受けないので局所的には正しく系の運動を追跡し、かつゆるやかに集団変数の張る空間を探索することが可能となります。仮想的な粒子の質量は、上記の原理より集団変数の固有振動モードが系よりも遅くなるように設定します。

Meta-dynamics 法は、仮想粒子の動力学を追跡するのではなく、系に直接バイアスポテンシャルを足しこんでいくことによって実現する手法もあります[3]。このような方針を採用すると、仮想的な粒子の質量やそれと集団変数を結びつけるばね定数の定義が不要となり、よりシンプルに実行することが可能となります。

5.4.3.2 入力パラメータ

本機能と関連あるタグの一覧を表 5.4 に示します。

表 5.4 メタダイナミクスに関連のあるタグの一覧

第1ブロック識別子	第2, 第3ブロック識別子	タグ識別子	説明
control			
		driver	ダイナミクスの種類を選択する変数。 拘束条件付きダイナミクスの場合 meta_dynamics を指定する。
meta_dynamics			メタダイナミクスの設定を行うブロック
		meta_dynamics_type	メタダイナミクスの“種類”の指定 bias_and_fictitious, bias_only, bias_generation n のいずれか デフォルト値は bias_only.
		max_bias_update	最大バイアス更新回数を指定する。 デフォルト値は1(負の数値の場合、この条件 では停止しないことを意味する)。
		extensive_output	on とすると通常は不要な出力も得られる
		output_per_rank	on とするとレプリカ並列時に各ランクごと に出力が得られる
	collective_variable		集団変数の指定を行うブロック。
		type	集団変数の“種類”の指定。拘束条件付きダイ ナミクスと同じ。 bond_length, bond_angle, dihedral_angle, bond_length_diff, bond_angle_diff, distance_from_pos, plane, center_of_mass, coordination_number
		atomx	拘束条件に関わる原子を指定する。x は整数 で、たとえば type = bond_length の場合 2 原子が拘束に関わるので atom1 と atom2 で 指定する。

		k	仮想粒子のばね定数を指定する
		delta_s	バイアスポテンシャルの幅, Δs を指定する
		smin	バイアスポテンシャル出力の最小値を指定
		smax	バイアスポテンシャル出力の最大値を指定
		ds	バイアスポテンシャル出力の刻み幅を指定
		control_velocity	on にした場合, 仮想粒子の速度を制御する
		mass_thermo	仮想粒子の速度を制御する場合の熱浴の質量
		target_KE	仮想粒子の速度を制御する場合のターゲット運動エネルギー
	plane		面内拘束における面の法線ベクトルを指定するブロック
		normx,normy,normz	法線ベクトルの x,y,z 成分
	distance_from_pos		場所の指定を行うブロック
		posx,posy,posz	指定したい場所の x,y,z 成分
	coordination_number		配位数指定を行うブロック
		kappa_inv	配位数定義式の κ の逆数を長さの単位で指定
		kappa	配位数定義式の κ を 1/bohr 単位で指定
		rcut	配位数定義式の r_c の値を長さの単位で指定
	center_of_mass		重心を変化させる方向を指定するブロック
		directionx	指定したい方向の x 成分
		directiony	指定したい方向の y 成分
		directionz	指定したい方向の z 成分
	bias_potential		バイアスポテンシャルの設定を行うブロック
		height	一度に足すバイアスポテンシャルの高さを
			エネルギーの単位で指定
		update_frequency	バイアスポテンシャルを足す頻度を指定 たとえば 10 とすると, 10MD ステップに 1 回バイアスポテンシャルが更新される。 デフォルト値は 20。
		output_frequency	meta_dynamics_type が bias_generation の場合に, バイアスポテンシャル構築を何回に 1 回行うかを指定する。 デフォルト値は 10
	continuation_strategy		レプリカ並列計算時における継続計算の方針を設定する。
		randomize_velocity	on とすると速度は継続せず乱数で決まるようになる
		scale_velocity	on とすると継続時下記の velocity_scaling_factor に応じて速度がスケールされる。
		velocity_scaling_factor	読み込んだ速度をスケールする値。デフォルト値は 1
		configuration_from_input	on とすると, 継続計算ファイルではなく入力ファイルから座標データを読み込む。デフォルト値は off

Meta-dynamics 法の入力パラメータの指定について説明します。

Meta-dynamics 法の計算は, 以下の指定を行います。

- Meta-dynamics 法を有効に指定する
- Meta-dynamics の振る舞いを指定する (ダイナミクス追跡モード, バイアスポテンシャル更新回数, 出力形式など)
- 温度一定の MD シミュレーションの設定を行う
- 集団変数の定義を行う (集団変数に含める反応座標の情報を集団変数の数だけ定義)
- バイアスポテンシャルの設定を行う (バイアスポテンシャルの高さ, 幅, 更新頻度など)
- レプリカ並列計算を実行する場合, その設定。
- Meta-dynamics 法を有効に指定する

Meta-dynamics 法の機能を有効にするには、control ブロックにおいて以下の指定を行います

```
control{
    driver = meta_dynamics
}
```

この指定により、PHASE の通常の原子ダイナミクスコードではなく、Meta-dynamics 計算用のメイン・プログラムが呼ばれます。

● Meta-dynamics の振る舞いの設定

Meta-dynamics の全体的な振る舞いの設定方法を説明します。この設定は、入力ファイルの最上位に meta_dynamics ブロックを作成し、さらに以下のような変数・ブロックを定義することによって行います。

```
meta_dynamics{
    meta_dynamics_type = bias_only
    max_bias_update = -1
    extensive_output=on
    output_per_rank=on
    output_cvar_every_step=off
    continuation_strategy{
        randomize_velocity=on
        scale_velocity=off
        velocity_scaling_factor=0.7
        configuration_from_input=off
        ...
        ...
    }
}
```

meta_dynamics ブロックでは、以下の変数/ブロックの設定を行うことができます。

meta_dynamics_type 変数

bias_and_fictitious, bias_only, bias_generation のいずれかを指定します。bias_and_fictitious とすると仮想粒子の動力学を考慮したメタダイナミクスを、bias_only とするとバイアスポテンシャルのみを有効にしたメタダイナミクスを実行することができます。bias_generation とした場合 Meta dynamics は実行されず、作業ディレクトリーに存在するファイルからバイアスポテンシャルの構築および出力のみが実行されます。

max_bias_update 変数

最大何回バイアスポテンシャルを更新するかを指定します。負の値を指定すると、バイアスポテンシャルの更新回数では計算は停止しません。これがデフォルトの振る舞いです。

output_per_rank 変数

n とすると、レプリカ並列計算実行時に出力が各ランクごとに得られます。

extensive_output 変数

on とすると、仮想粒子の速度や仮想粒子に働く力など、通常は不要な出力も得られます

continuation_strategy ブロック

レプリカ並列計算実行時における継続計算の方針の設定を行います。レプリカ並列計算時には、並列数を変化させた場合に以前の計算を厳密に再現することはできないので、ここでどのような方針で継続するのか決定する必要があります。このブロックでは、以下の設定を行うことができます。

randomize_velocity

on とすると、継続計算実行時に速度を継続せず、乱数で割り振ります。デフォルト値は off。

scale_velocity

on とすると、読みこんだ速度をつぎに説明する

	velocity_scaling_factor の値に応じてスケールします。デフォルト値は off。
velocity_scaling_factor	読み込んだ速度にここで指定した値をかけます。デフォルト値は 1。
configuration_from_inpu	on とすると、継続計算ファイルではなく入力ファイルから座標データを読み込む。デフォルト値は off。

● 集団変数の定義

集団変数とは、“反応座標”を複数まとめたものです。この指定は、meta_dynamics ブロック以下において行います。典型的な例は下記の通りです。

```
meta_dynamics{
  ....
  ....
  collective_variable{
    mass=1000
    k=100
    delta_s = 0.08
    control_velocity=on
    mass_thermo = 50
    target_KE = 0.1
  }
  collective_variable1{
    type=bond_length
    atom1=5
    atom2=4
    delta_s=0.05 angstrom
    smin=1 angstrom
    smax=5 angstrom
    ds = 0.1 angstrom
  }
  ....
  ....
}
```

まず、meta_dynamics ブロックの下に collective_variable ブロックを作成します。collective_variable ブロックには全集団変数に共通の設定を行います。後に説明する集団変数個別の設定に記述がない項目のみここでの設定が反映されます。

次に、集団変数を定義したい数だけ collective_variablexx ブロックにおいて定義します。ここで xx は集団変数の ID です。任意の数の集団変数を定義することが可能ですが、1 から連続的に変化する整数を指定する必要があります。たとえば、collective_variable1、collective_variable2、collective_variable4 の 3 つの collective_variablexx ブロックがあった場合、collective_variable1 と collective_variable2 のみ解釈されます。collective_variable および collective_variablexx ブロックは、拘束条件付きダイナミクスの設定の際に設定する拘束条件と同様の変数を定義することが可能となっています。具体的には、以下の変数を定義することが可能です。

type 変数		集団変数の“種類”を指定します。以下のいずれかの値をとります。
	bond_length	2 原子間の距離を集団変数とします。
	bond_angle	3 原子の成すボンド角を集団変数とします。
	dihedral_angle	4 原子の 2 面角を集団変数とします。
	bond_length_diff	2 原子間の距離の差を集団変数とします。
	plane	ある原子の指定の面内での位置集団変数とします。
	center_of_mass	指定の原子群の重心を位置集団変数とします。

	coordination_number	配位数を位置集団変数とします。
	distance_from_pos	ある場所からの距離を集団変数とします。
atomx 変数		指定の集団変数に関わる原子を指定します。x は数字であり、たとえば 2 原子間の距離の場合は 2 つの原子が拘束に関わるので、atom1 と atom2 に対応する原子の番号を指定します。type が coordination_number の場合、配位数を計算する中心の原子の番号を指定します。
plane ブロック		面内拘束の場合の、拘束したい面の法線ベクトルを指定するためのブロックです。次の変数を指定することができます。
	normx	法線ベクトルの x 座標。
	normy	法線ベクトルの y 座標。
	normz	法線ベクトルの z 座標。
coordination_number ブロック		配位数拘束の場合原子 i の配位数 n_i は $n_i = \sum_{j \neq i} \frac{1}{\exp[\kappa(r_j - r_i - r_c)] + 1}$ と定義していますが、この式における κ, r_c の値を指定するブロックです。次の変数を指定することができます。
	kappa_inv	$\frac{1}{\kappa}$ の値を、長さの単位で指定します。
	kappa	κ の値を、1/bohr 単位で指定します。kappa_inv よりも優先されます。
	rcut	r_c の値を長さの単位で指定します。
mass 変数		仮想粒子の質量を指定します。meta_dynamics_type が bias_and_fictitious の場合のみ意味のある指定です。
k 変数		仮想粒子と集団変数の結びつきを決める、ばね定数を指定します。meta_dynamics_type が bias_and_fictitious の場合のみ意味のある指定です。
delta_s 変数		(23)式の δs_q の値を指定します。
smin 変数		バイアスポテンシャル出力の際の最小値を指定します。
smax 変数		バイアスポテンシャル出力の際の最大値を指定します。
ds 変数		バイアスポテンシャル出力の際の“刻み幅”を指定します。
control_velocity 変数		“on”にした場合、仮想粒子のダイナミクスを追跡する際に熱浴を付与することによってその速度を制御します。meta_dynamics_type が bias_and_fictitious の場合のみ意味のある指定です。
mass_thermo 変数		control_velocity が“on”の場合の、熱浴の“質量”を指定します。
target_KE 変数		control_velocity が“on”の場合の、目的とする仮想粒子の温度を指定する。

● バイアスポテンシャルの設定

バイアスポテンシャルの設定は、meta_dynamics ブロックの下に bias_potential ブロックを作成して行います。以下に典型的な例を示します。

<pre> bias_potential{ height = 0.02 eV update_frequency=20 output_frequency=100 }</pre>

bias_potential ブロックにおいて定義可能な変数は下記の通りです。

height 変数	一度に足すバイアスポテンシャルの高さをエネルギーの単位で指定します。一度に足すバイアスポテンシャルの幅は、各集団変数固有の量であるので bias_potential ブロックではなく集団変数固有の設定を行う collective_variablexx ブロック以下で行います。
output_frequency 変数	meta_dynamics_type が bias_generation の場合のみ意味のある指定です。何回に 1 回バイアスポテンシャルを出力するかを指定します。
update_frequency	バイアスポテンシャルの更新頻度を指定します。デフォルト値は 20 です。

● レプリカ並列計算の設定

初期速度を変化させる

特に指定がない場合、初期座標はすべてのレプリカで共通で、初期速度の乱数のみ異なる、という条件で計算がなされます。位相空間上異なる点から始めるので、座標が同じでも各レプリカはいずれ異なる軌跡をとります。ただし、当然のことながら最初のうちは（座標値は）ほぼ同じ軌跡となります。

初期の座標値をランクごとに明示的に指定する

入力ファイルにおいて、レプリカごとに異なる座標データを指定することも可能です。この設定は、**atomsxx** ブロック（ここで **xx** は MPI ランクの数字）を作成し、そこで座標値を設定することによって行います。たとえば、ランク 0 が担当するレプリカとランク 1 が担当するレプリカにそれぞれ異なる座標値を与えるには、次のような記述を行います。

```

structure{
  atom_list{
    ....
    atoms0{
      #units angstrom
      #default weight = 1, element = Si, mobile = 1
      #tag element rx ry rz mobile weight
      C 5.0157363043      5.6563796505      5.8043454319 1 1
      C 4.7499007526      4.2727134018      5.7364572058 1 1
      ...
      ...
    }
    atoms1{
      #units angstrom
      #default weight = 1, element = Si, mobile = 1
      #tag element rx ry rz mobile weight
      C      4.5897384578      5.5998560107      5.7723226564 1 1
      C      5.1658344359      4.3217914066      5.6857269157 1 1
      ...
      ...
    }
  }
}

```

5.4.3.3 計算の実行方法

Meta-dynamics 法を実行するには、通常の PHASE による計算と同様に以下のコマンドを発行します。

```
mpirun -n NP phase ne=NE nk=NK nr=NR
```

ここで NP は MPI プロセス数、NE はバンド並列数、NK は k 点並列数、NR はレプリカ並列数です。NP=NE×NK×NR という関係が成立している必要があります。ne, nk, nr はいずれも省略可能（デフォルト値は 1 すべて無指定の場合 NE=NP）です。

通常 Meta dynamics 実行時に得られるバイアスポテンシャルの出力は「直近に得られたバイアスポテンシャル」のみですが、バイアスポテンシャルをポスト処理的に計算し、それを出力させることもできます。この機能を利用するには、入力ファイルの meta_dynamics ブロックの meta_dynamics_type 変数に bias_generation という文字列を指定します。この時、meta_dynamics ブロックの下に bias_potential ブロックにおいて定義される、bias_output_frequency 変数に指定された回数に 1 回出力を行います。たとえば、bias_output_frequency が 10、バイアスの総更新回数が 100 だった場合、10 回目、20 回目、30 回目、... 100 回目の更新時のバイアスポテンシャルがそれぞれ独立したファイルに出力されます。そのファイル名は、“bias_potential.dataxx”となります。ここで xx が対応する更新回数です。この設定を行ったあと、Meta-dynamics 解析を行ったディレクトリーにおいて PHASE を実行します。ファイルを読み込みバイアスポテンシャルを構築するのみなので、通常並列で実行する必要はありません。

5.4.3.4 計算結果の出力

Meta dynamics シミュレーションを行う場合、標準よりも多くのファイルが出力されます。以下に、各々について簡単に説明します。

- curr_bias_potential.data ファイル

“現在の”バイアスポテンシャルが記録されたファイルです。次のような形式で出力されます。

1.2000000000	-3.1400000000	0.0000000000
1.3000000000	-3.1400000000	0.0000000000
1.4000000000	-3.1400000000	0.0000000000
1.5000000000	-3.1400000000	0.0000000000
1.6000000000	-3.1400000000	0.0000000000
1.7000000000	-3.1400000000	0.0000000000
....		
....		
1.2000000000	-3.0400000000	0.0000000000
1.3000000000	-3.0400000000	0.0000000000
1.4000000000	-3.0400000000	0.0000000000
1.5000000000	-3.0400000000	0.0000000000
1.6000000000	-3.0400000000	0.0000000000
1.7000000000	-3.0400000000	0.0000000000
....		
....		

各行が“集団変数の組”に相当します。定義している数だけ集団変数が記録されたあと、その“集団変数の組”におけるバイアスポテンシャルの値が出力されます。

- bias_potential.dataxx ファイル

バイアスポテンシャルを作成するのみのモードを利用した場合に得られる、更新回数に応じたバイアスポテンシャルのデータが出力されるファイルです。ファイル名の xx がバイアスポテンシャル更新回数に相当します。そのファイル形式は、curr_bias_potential.data と同様です。

- nfdynm.data_at_bias ファイル

バイアスポテンシャル更新時における座標データが出力されるファイルです。PHASE の標準座標データ出力形式である、F_DYNM 形式で出力されます。

- nfefn.data_at_bias ファイル

バイアスポテンシャル更新時におけるエネルギーの値が出力されるファイルです。PHASE の標準的なエネルギーデータ出力形である、F_ENF 形式で出力されます。

- collective_variables.data ファイル

バイアスポテンシャル更新時における集団変数の値が出力されるファイルです。次のような形式で出力されます。

2	1.6399047278	0.0906233310
3	1.6933783940	0.2327954221
4	1.6487636847	0.0655806009
5	1.7510381463	-0.1403803460
6	1.7880912692	-0.2122517967
7	1.7558411086	-0.2557274737
8	1.7939362867	-0.0296094373
9	1.7595919709	0.1959354384
10	1.7773637731	0.3761827029
11	1.7657919080	0.3998392061
12	1.7604309483	-0.0107912799
13	1.6218441177	-0.3366407543

各行がバイアスポテンシャル更新のタイミングに対応します。一列目がバイアスポテンシャルの更新回数であり、二列目以降定義順に対応する集団変数の値が出力されます。

- bias_potential_parameters.data ファイル

バイアスポテンシャルのパラメーターが出力されるファイルです。継続計算のタイミングでこのパラメーターを変化させた場合、それ以前のパラメーターの値が分らないとバイアスポテンシャルの構築ができないことから必要なファイルです。次のような形式で出力されます。

2	0.0200000000	0.1000000000	0.1000000000
3	0.0200000000	0.1000000000	0.1000000000
4	0.0200000000	0.1000000000	0.1000000000
5	0.0200000000	0.1000000000	0.1000000000

各行がバイアスポテンシャル更新のタイミングに対応します。一列目がバイアスポテンシャル更新回数であり、二列目が(23)式における w の値、3列以降が各集団変数の(23)式における δs_α の値です。

5.4.3.5 計算例：炭化水素のエネルギー表面

(1) 概要

Meta dynamics 法を利用した例として、炭化水素のエネルギー表面を調べた例を紹介します。具体的には、 C_4H_6 分子の電子環状反応を取り上げます。 C_4H_6 分子は、trans 1-3 ブタジエン、cis 1-3 ブタジエン、シクロブテンの3種類の安定構造が知られています。シクロブテンは環状分子、trans 1-3 ブタジエンは平面状の分子ですが、cis 1-3 ブタジエンは平面状にはならず、2面角を30°ほどひねった構造が安定な構造です(gauche 配座)。その分子構造を図 5-57 に示します。エネルギーは、高い順にシクロブテン、cis 1-3 ブタジエン、trans 1-3 ブタジエンであり、分子の反応としては、1-3 ブタジエンが閉環して環状化合物であるシクロブテンを生成する、あるいは逆にシクロブテンが開環し1-3 ブタジエンが生成される反応(電子環状反応)、また、2種類の1-3 ブタジエンの間の cis-trans 反応が考えられます。閉環・開環反応は化学結合の切断を要することから大きな障壁エネルギーがあり、1 eV 程度のオーダーであると考えられます。他方、cis から trans への変化はそこまでの障壁はなく、100 meV 程度のオーダーであると考えられます。特に、環状反応においては、1-3 ブタジエンとシクロブテンとでは2重結合の数が異なり、電子状態としては全く異なるものであるため、古典的なポテンシャルで取り扱うのは一般に難しいと言えます。この点を PHASE で正しく扱えるかどうかを確認します。

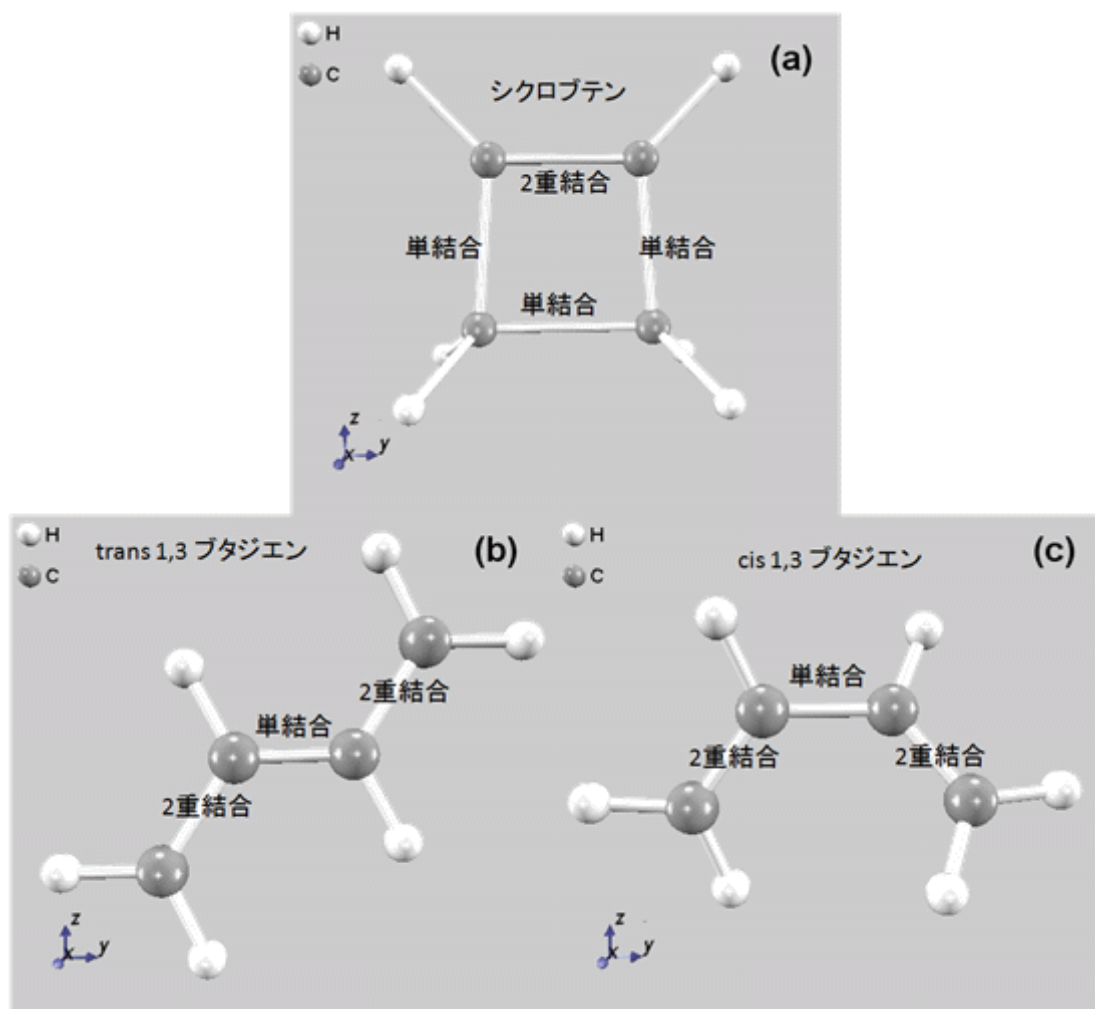


図 5-57 C_4H_6 分子の分子構造

初期の原子配置は、図 5-58 で示すシクロブテンを採用します。この初期構造は、PHASE によって最適化したものです。

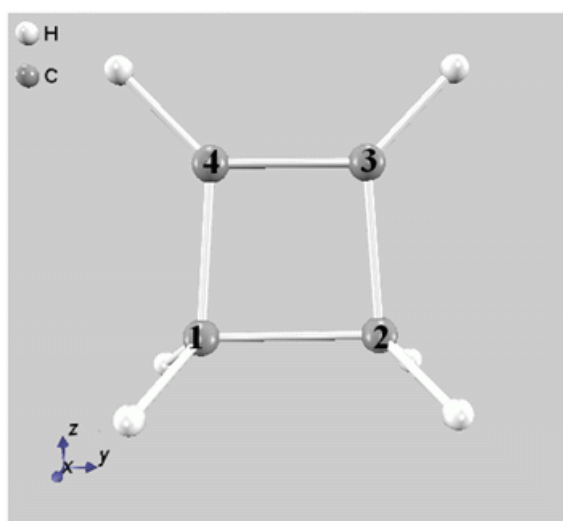


図 5-58 C_4H_6 分子の分子構造

(2) 入力パラメータ

Meta dynamics 法を有効に指定します。これは、control ブロックの下で driver 変数に meta_dynamics を指定することによって行います。

```
condition{
    driver = meta_dynamics
    ....
}
```

次に集団変数を定義します。その方針は様々ですが、ここでは以下を採用します。

1. 図 24 の原子1と原子2の距離。パラメーターds, delta_sはそれぞれ0.1 Åと0.05 Å
2. 図 24 の, 原子1-4-3-2の作る二面角。パラメーターds, delta_sはそれぞれ10°と5°

この設定は、meta_dynamics ブロックの下で以下のように実現します。

```
meta_dynamics{
    ....
    ....
    collective_variable1{
        type=bond_length
        atom1=5
        atom2=4
        delta_s=0.05 angstrom
    !for bpot output
        smin=1 angstrom
        smax=5 angstrom
        ds = 0.1 angstrom
    }
    collective_variable2{
        type=dihedral_angle
        atom1=5
        atom2=3
        atom3=2
        atom4=4
        delta_s = 5 degree
    !for bpot output
        smin = -180 degree
        smax = +180 degree
        ds = 10 degree
    }
}
```

バイアスポテンシャルの高さは 0.02 eV(0.46 kcal/mol)とします。バイアスポテンシャルの更新頻度は、20 MD ステップに一度とします。この設定は、meta_dynamics ブロックの下に bias_potential ブロックを作成し、height パラメーターで指定することによって行います。

```
meta_dynamics{
    ....
    ....
    bias_potential{
        update_frequency = 20
        height=0.02 eV
    }
}
```

バイアスポテンシャルを更新する回数は任意ですが、信頼できる自由エネルギー表面を得るためには相当数の更

新回数が必要です。

(3) 計算結果

本シミュレーションによって得られる計算結果を解説します。まず、図 5-59 にバイアスポテンシャルを 18,140 回程度更新した結果得られたエネルギー表面の等高線図を示します。

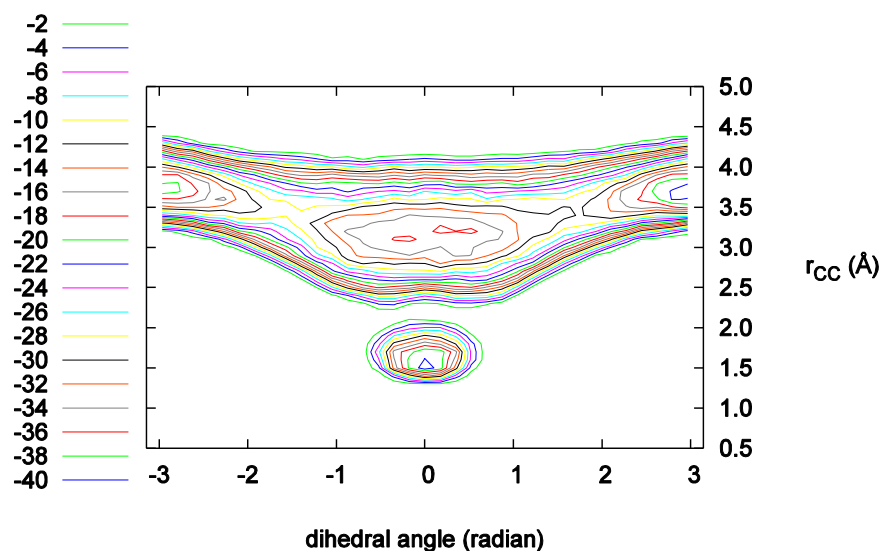


図 5-59 C₄H₆分子の自由エネルギー表面

図 5-59 より、ここで得られたエネルギーの等高線図には 4 つの安定点があることが理解できます。すなわち、原子間距離が約 1.5 Å 程度で角度がほぼ 0 radian の点、原子間距離が 3.3 Å 程度で角度が 0 radian、原子間距離が 3.7 Å 程度で角度が ±3 radian 程度の 2 つの点です。これらは、それぞれシクロブテン、cis 1-3 ブタジエン、trans 1-3 ブタジエンに相当します。絶対零度の計算の場合、cis ではなく gauche 配座となりますが、300K の Meta dynamics シミュレーションでは cis と gauche の明確な区別がつけられる結果は得られませんでした。得られたシクロブテンと trans 1-3 ブタジエンのエネルギー差は、16 kcal/mol 程度、シクロブテンと cis ブタジエンのエネルギー差は 12 kcal/mol 程度となりました。いずれも、絶対零度の計算と比較するとより大きなエネルギー差です。

図 5-60 および図 5-61 には、集団変数がバイアスポテンシャルの更新と共にどのように変化していったかを示しています。二面角が図 26、炭素原子間距離が図 27 の振る舞いです。図 5-60 および図 5-61 より、バイアスポテンシャルを約 700 回更新した時点で鞍点を超えてブタジエンに至っていることが理解できます。そこから 18,000 回程度の更新までは幅広くエネルギー表面を探索しています。図 5-57 らも分かるように、ここで考えている系はシクロブタンを除くと二面角に対して幅広い範囲の構造を取り得ます。そのため、この谷を埋め尽くすのに多くのバイアスポテンシャルの更新が必要となっています。18,000 回程度のバイアスポテンシャル更新の結果、再びシクロブタンへ戻ったことが確認できた時点(図 5-62)で計算を終了させました。

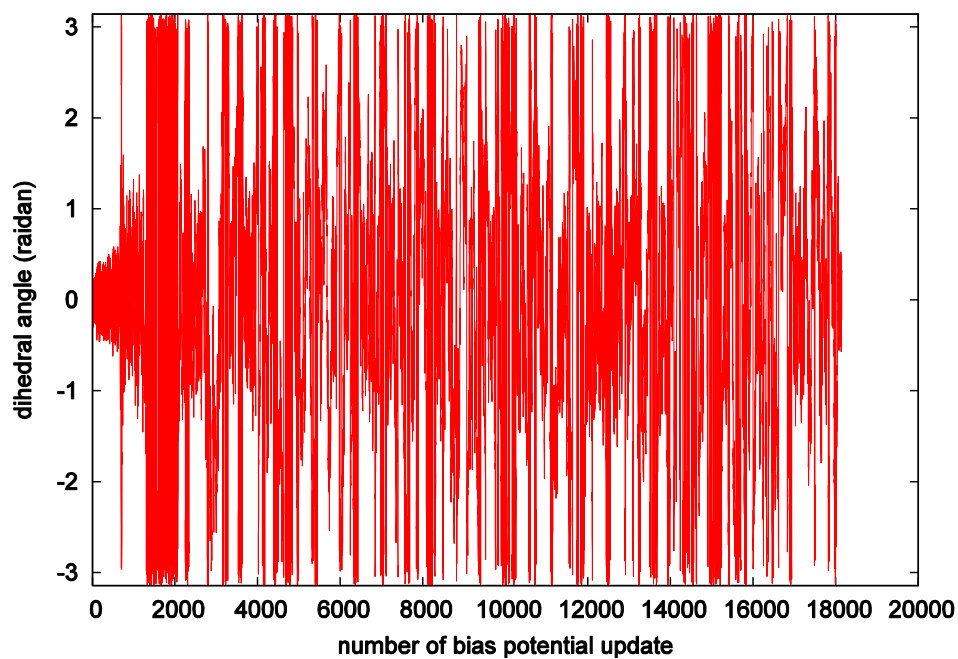


図 5-60 2 面角とバイアスポテンシャル更新回数の関係

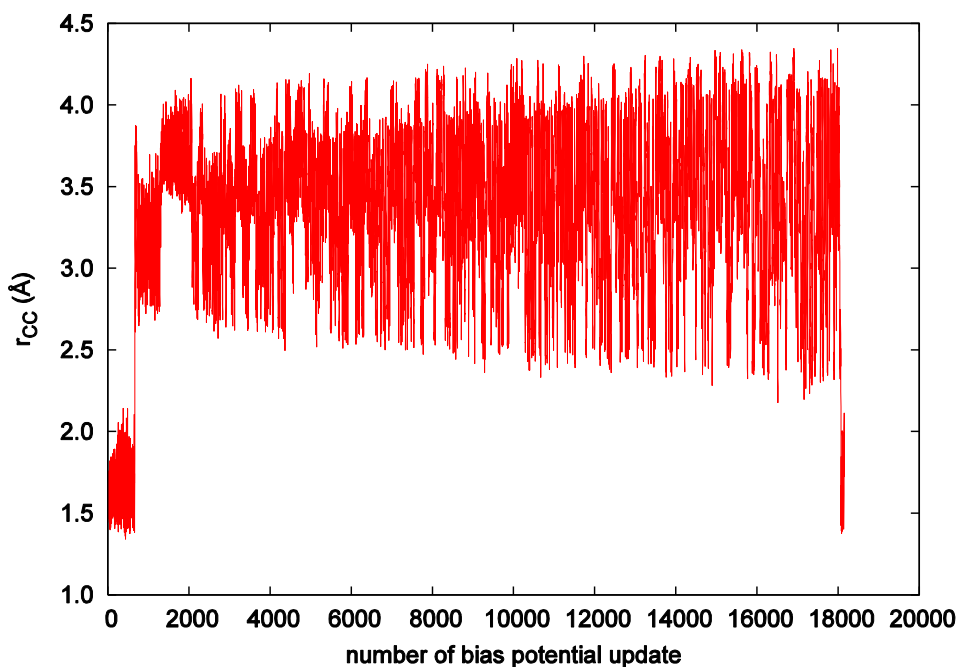


図 5-61 炭素原子間距離とバイアスポテンシャル更新回数の関係

図 5-62 (a)から(d)までに, Meta dynamics シミュレーション中に実際に得られた原子配置のスナップショットを示しました。ここで示しているように, バイアスポテンシャルの効果によって様々な分子構造が実現していることが分かります。

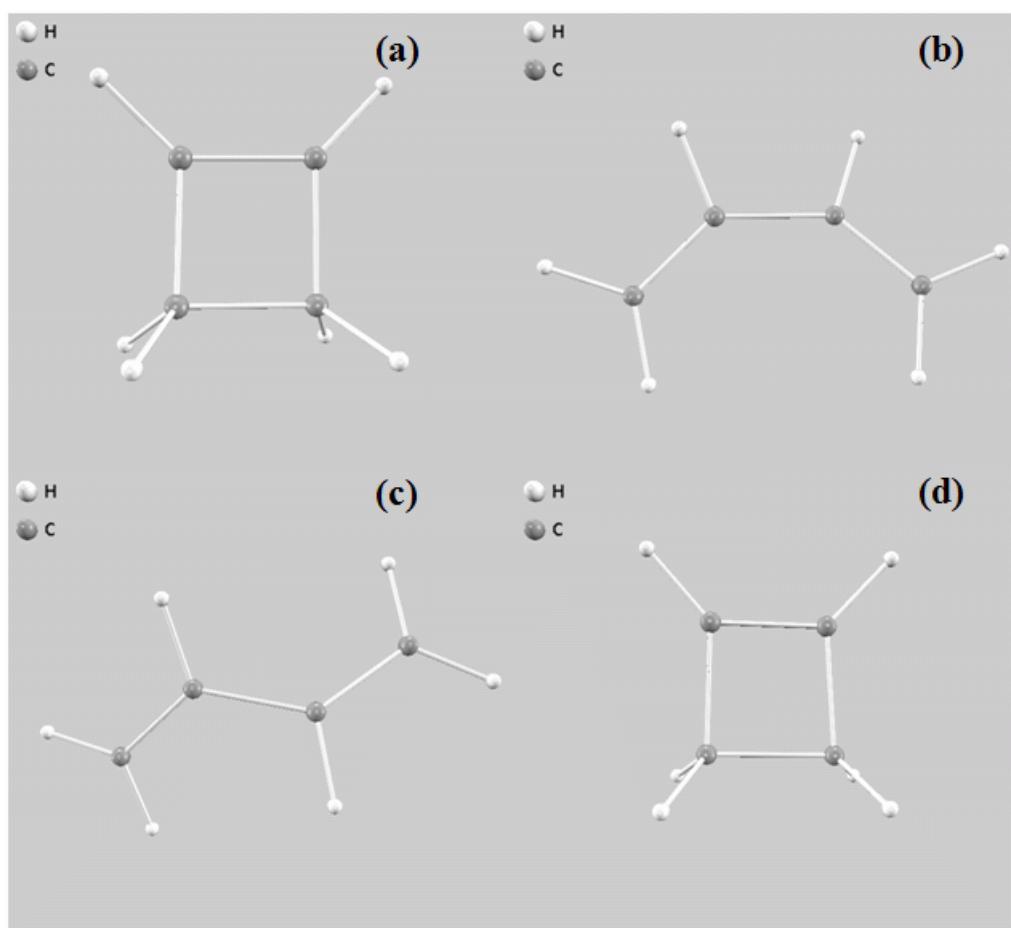


図 5-62 Meta dynamics シミュレーションによって得られた分子構造のスナップショット. (a): バイアスポテンシャル 2 回更新 (b) バイアスポテンシャル 690 回更新 (c) バイアスポテンシャル 1,500 回更新 (d) バイアスポテンシャル 18,070 回更新

5.4.3.6 使用における注意点

Meta dynamics 法は、すべての擬ポテンシャルと組み合わせて利用することができます。レプリカ並列を含めた並列計算も行うことができます。ただし、意味のある結果を得るためには膨大な計算量を費やす必要があります。レプリカ並列を行う場合、継続計算のタイミングでレプリカ並列数を変化させる場合、対応するレプリカの継続計算ファイルが存在しない場合があります。この場合は近くのランクの継続計算データを読み込み、さらに `continuation_strategy` で設定した指針に従って初期レプリカを作成します。

5.4.3.7 参考文献

- [1] A. Laio and M. Parrinello, Proceedings of the National Academy of Sciences **99**, (2002) p. 12562.
- [2] M. Iannuzzi, A. Laio and M. Parrinello, Physical Review Letters **90**, (2003) p. 238302.
- [3] A. Laio, A. Rodriguez-Forteza, F. L. Gervasio, Ceccarelli and M. Parrinello, J. Phys. Chem. B **109**, (2005) p. 6714.

5.5 時間依存密度汎関数法計算 (TDDFT)

5.5.1 実時間処理型の時間依存密度汎関数理論 (RTTDDFT) による光学スペクトル計算

5.5.1.1 機能の概要

実時間 TDDFT 法に基づいて、次式の時間依存一電子方程式を解くことにより、与えられた初期一電子波動関数に対する電子ダイナミクス・シミュレーションを行う。

$$i\hbar \frac{\partial}{\partial t} \phi_n^k(\mathbf{r}, t) = H(t) \phi_n^k(\mathbf{r}, t)$$

波数ベクトル \mathbf{k} とバンド番号 n を指標とする一電子波動関数を ϕ_n^k で、 ϕ_n^k 有効ハミルトニアンを H で表す。時間依存一電子方程式の形式的解から、一電子波動関数の時間発展は以下のように表される。

$$\phi_n^k(\mathbf{r}, t + \Delta t) = \exp\left(-\frac{i}{\hbar} \int_t^{t+\Delta t} dt' H(t')\right) \phi_n^k(\mathbf{r}, t)$$

右辺の時間発展演算子は、時間積分部および指数関数部に対して近似を施して数値計算され、それぞれ様々な近似方法が提案されている。時間積分部は時間間隔 Δt が十分に小さければ次式のように近似できる。

$$\phi_n^k(\mathbf{r}, t + \Delta t) \cong \exp\left(-\frac{i}{\hbar} \Delta t H(t)\right) \phi_n^k(\mathbf{r}, t)$$

さらに、指数関数部に対する近似としてテイラー展開法が使用される。

$$\exp\left(-\frac{i}{\hbar} \Delta t H(t)\right) = \sum_{N=0}^{\infty} \frac{1}{N!} \left(-\frac{i}{\hbar} \Delta t H(t)\right)^N$$

この近似法の場合、時間間隔 Δt と展開最大次数 N_{\max} が主要な入力パラメータであり、計算精度と実行時間の兼ね合いを考慮して最適値を設定する必要がある。

原理的には、各時間においてヘルマン・ファインマン力を計算し、その値に従って原子位置のダイナミクスを実行することも可能である。(ただし、本プログラムには実装されておらず、原子位置は固定した状態で電子ダイナミクス計算を行う。)

時間 $t=0$ の初期一電子波動関数は、プログラム PHASE によって求められる基底状態の一電子波動関数を用いて作成するため、本機能実行前に DFT 法による基底状態の計算を終えている必要がある。本プログラムでは、基底状態一電子波動関数 $\phi_n^k(\mathbf{r}, t=0)$ を以下のように位相シフトさせることができる。

$$\phi_n^k(\mathbf{r}, t = 0^+) = e^{-i\epsilon \mathbf{q} \cdot \mathbf{r}} \phi_n^k(\mathbf{r}, t = 0^-)$$

これは時間 $t=0^+$ においてパルス電場を系に与えることに相当する。各時刻において双極子モーメント $\mathbf{d}(t)$ または電流密度 $\mathbf{J}(t)$ を計算し、電子ダイナミクス・シミュレーション終了後に、これらの値を時間の関数から振動数へとフーリエ変換することによって、実験と直接比較可能な双極子強度・光吸収スペクトルなどの光学物性値が求められる。

5.5.1.2 入力パラメータ

以下の入力タグの例は、初期状態として「x 方向へ大きさ 0.01 原子単位 of デルタ関数型パルス電場」を系に与え、「時間刻み 0.1 原子単位時間/時間ステップで 1,000 時間ステップ (全シミュレーション時間 100 原子単位時間)」の RTTDDFT 計算を実行する場合です。ただし、DFT 法による基底状態の計算が収束している場合のみ RTTDDFT 計算が実行されます。

postprocessing{

```

rttdfft{
  sw_rttdfft = on
  time_step_delta = 0.1
  time_step_max = 1000
  ext_pulse_epsilon = 0.01
  ext_pulse_kx = 1.0
  ext_pulse_ky = 0.0
  ext_pulse_kz = 0.0
}
}

```

パラメータ	デフォルト値	説明
sw_rttdfft	OFF	ON : RTTDDFT 計算処理を実行する OFF : RTTDDFT 計算処理を行わない
time_step_delta	0.1 (原子単位時間)	時間更新刻み 全シミュレーション時間は、time_step_delta×time_step_max となる。 経験的には 0.05～0.1 原子単位時間の値が推奨される (1 原子単位時間はおよそ 0.024 fs) 大きな time_step_delta を設定した場合には異常値が出力されることがあり、その場合は値をより小さく設定して再計算する必要がある。
time_step_max	100 (回)	計算処理を終了するまでの時間更新回数 [計算途中にジョブを正常終了させることはできない (nfstop 機能は使用不可)]

初期状態生成 [デルタ関数型パルス電場 $E(t) = E_0 e \delta(t)$] に関するパラメータ

パラメータ	デフォルト値	説明
ext_pulse_epsilon	0.0d0	電場の大きさ E_0
ext_pulse_kx	0.0d0	電場の向き e
ext_pulse_ky	0.0d0	電場の向き e
ext_pulse_kz	0.0d0	電場の向き e

5.5.1.3 計算の実行方法

まずは、通常の SCF 計算を行います。この際は、sw_rttdfft パラメータは off に設定しておきます。通常の SCF 計算が終了したら、sw_rttdfft パラメータを on とし、さらに必要に応じて rttdfft ブロックの各種パラメータを編集し、control ブロックの下で condition パラメータを continuation に設定します。このような設定を施したら、通常通り PHASE の計算を実行すると TDDFT 計算が実行されます。途中で終了した RTTDDFT 計算は、初期電場の設定 (ext_pulse_epsilon, ext_pulse_kx, ext_pulse_ky, ext_pulse_kz) をすべて 0 とし、condition=continuation としたままで PHASE を実行すれば行うことができます。

5.5.1.4 計算結果

計算結果は、ログファイルに記録されます。以下のように、双極子モーメントと電流密度が原子単位で記録されます。

```

# time_step= 991 time= 0.9910E+02 au = 0.2397E+01 fs
...
P=      0.0285381798      0.0002058360      0.0001702915
J=     -0.0133497494      0.00000068680     -0.0000030064

```

P=のあとの 3 つの数値が双極子モーメントの x, y, z 成分、J=のあとの 3 つの数値が電流密度の x, y, z 成分です。

計算終了後、時間空間から周波数空間にフーリエ変換することによってスペクトルをエネルギーの関数として得ることが可能です。このフーリエ変換を実行するプログラムのソースコードが `src_spectrum` の下にある `spectrum.f90` です。このプログラム自体は、標準的な Fortran90 コンパイラによってコンパイルすることが可能です。その利用方法は、以下の通りです（作業ディレクトリーとして、`ft` というディレクトリーを作成して実行する例）。まず、以下のようにディレクトリー`ft` を作成し、そのディレクトリーの下に電流密度の履歴を抽出した `j.data` というファイルを作成します。

```
% mkdir ft
% cd ft
% grep "J=" ../output001 > j.data
```

`j.data` ファイルに、次のように 1 行目に `time_step_delta` と `ext_pulse_epsilon` を加え、さらに `time_step_max` の値を 2 行目に加えます。

```
0.1 0.01
1001
J=      -0.2999128661      0.0000002015      -0.0000008972
J=      -0.2970981909     -0.0000022160      -0.0000000155
...
...
```

このファイルを作成したら、`spectrum.f90` をコンパイルしたバイナリー（たとえば `a.out`）を実行します。結果としては、以下のファイルが得られます。

出力ファイル名	説明
<code>j.out</code>	電流密度と時間の関係、 $J_x(t)$ $J_y(t)$ $J_z(t)$
<code>p.out</code>	誘起された双極子モーメントと時間の関係、 $P_x(t)$ $P_y(t)$ $P_z(t)$
<code>pw.out</code>	フーリエ変換された双極子モーメント、 $\text{Re}[P_x(w)]$ $\text{Im}[P_x(w)]$
<code>abs.out</code>	双極子モーメント強度のスペクトル

5.5.1.5 例題

例として、ベンゼン分子（図 5-63）の RT-TDDFT 計算例を紹介します。この例題の入力ファイルは `samples/tddft/work_benzene` 以下にあります。

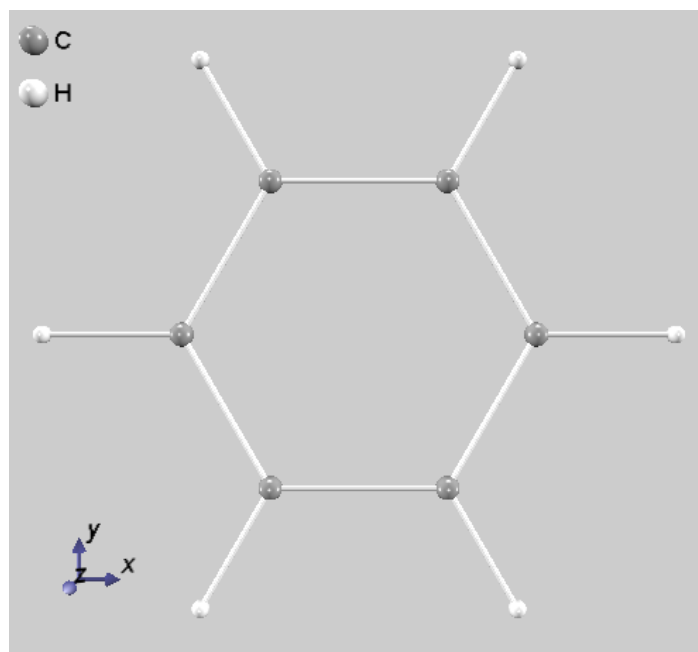


図 5-63 ベンゼン分子の原子配置

計算は、まずは SCF 計算を実行し、ついで RT-TDDFT を有効にした継続計算を実行しました。RT-TDDFT のステップ数は 11,000 とし、時間刻みは 0.1 au (約 0.0024 fs) としました。さらに、得られた結果を 5.5.1.4 で説明した手続きによってフーリエ変換しました。得られる吸収スペクトル(abs.out の結果)は図 5-64 で示す通りです。

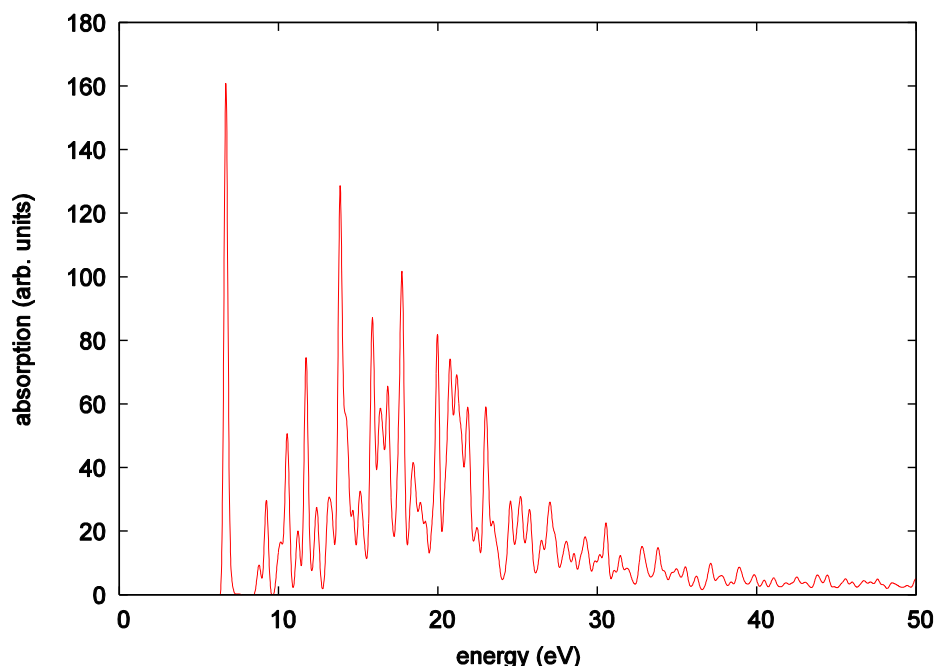


図 5-64 本例題によって得られる吸収スペクトル

最初のピークが、約 6.8 eV に表れています。通常の SCF 計算によって得られる HOMO-LUMO ギャップは約 5.1 eV ですが、この値よりも大きな値が吸収端として得られており、もっともらしい結果と考えられます。

5.5.1.6 使用上の注意

- ノルム保存型擬ポテンシャルを使用してください。ウルトラソフト型擬ポテンシャルには対応していません。

- 分子の重心がユニットセル中心に位置するように原子座標を設定し、分子が十分な真空領域に囲まれたユニットセルサイズを設定して下さい。（分子がセル境界をまたいで存在する場合には、双極子モーメントが正しく計算されません。）
- `ksampling{}`タグ内で「`base_reduction_for_GAMMA=off`」と「`base_symmetrization_for_GAMMA=off`」を明示して下さい。
- `symmetry{}`タグ内で「`method = manual`」と「`sw_inversion = off`」を明示して下さい。
- バルク系の計算には対応していません。

5.6 構造最適化

5.6.1 ストレステンソルを利用したユニットセル最適化機能

PHASE/0 には、ストレステンソルを利用して単位胞を最適化する機能が備わっています。ここでは、この機能の使い方の説明を行います。

5.6.1.1 入力パラメータ

まずは、通常通り入力パラメータファイルを記述します。セルを変形させたあとに座標の緩和を行いたい場合は通常通り原子座標最適化用のパラメータを設定すればセルの変形→力が収束していない場合は原子座標の最適化、という動作をするようになります。さらに、単位胞最適化用の、以下のような設定を加えます。

```
structure_evolution{
  lattice{
    sw_optimize_lattice = on
  }
}
```

変数 `sw_optimize_lattice` を `on` とすると本機能を利用することができます。 `lattice` ブロックには、以下の変数を定義することが可能です。

<code>sw_optimize_lattice</code>	単位胞最適化機能を有効にする場合 <code>on</code> とします。デフォルト値は <code>off</code> です。なお、このスイッチが <code>on</code> の場合は <code>sw_stress</code> は自動的に <code>on</code> になります。
<code>sw_uniform</code>	単位胞を一様に変化させたい場合に <code>on</code> とします。デフォルト値は <code>off</code> です。このパラメータが <code>on</code> の場合、ストレステンソルの対角要素の平均値によって体積を変化させるように動作します。
<code>sw_rebuild_pws</code>	単位胞を変形させた後に平面波基底を作り直すかどうかを指定します。デフォルト値は <code>on</code> 、つまり格子が変形する度に平面波を作り直します。 <code>Off</code> とすることによって電子状態計算の収束性を向上させることができますが、格子が変形しても同じ平面波セットを利用している、ということは厳密にはカットオフエネルギーが微妙に変化している、ということに相当する点に注意が必要です。また、このパラメータを <code>off</code> とすると継続計算ができなくなってしまいます。
<code>method</code>	最適化の手法を選択します。 <code>bfgs</code> , <code>quench</code> , <code>sd</code> のいずれかを指定します。デフォルト値は <code>bfgs</code> です。
<code>delta_stress</code>	<code>method</code> が <code>quench</code> か <code>sd</code> の場合の更新の刻み幅を指定します。デフォルト値は 1 です。
<code>max_stress</code>	収束判定に利用する、ストレステンソルの最大値を圧力の単位で指定します。デフォルト値は <code>1.e-6 hartree/bohr³</code> です。 <code>sw_uniform</code> が <code>on</code> の場合はストレステンソルの対角要素の平均が収束判定に採用されます。
<code>sw_optimize_coordinates_once</code>	原子配置の最適化は 1 回目の格子の更新時のみ行いたい場合に <code>on</code> とします。

なお、後述するようにストレステンソルのカットオフエネルギーに対する収束性はかなり悪い場合があります。ストレスミニマムと全エネルギーのミニマムが一致しない場合、おそらくカットオフが不十分であることが要因と思われます。このようなケースに遭遇したら、ストレステンソルとカットオフエネルギーの関係をしらべていただくことを推奨します。

5.6.1.2 計算結果の出力

結果は `output000` ファイル、`nfefn.data` ファイル、`nfdynm.data` ファイルに記録されます。

`output000` ファイルには、ストレステンソルが記録されます。以下のコマンドによってその情報を抽出することができます。

```
% grep -A3 'STRESS TENSOR$' output000
```

```

STRESS TENSOR
  0.0002326236      0.0000000000      0.0000000000
  0.0000000000      0.0002326236      0.0000000000
  0.0000000000      0.0000000000      0.0002142790
--
STRESS TENSOR
  0.0002272841      0.0000000000      0.0000000000
  0.0000000000      0.0002272841      0.0000000000
  0.0000000000      0.0000000000      0.0002077216
--
.....
.....

```

通常の計算の場合ストレステンソルが1組出力されるのみですが、本機能を利用している場合はストレステンソルの履歴が出力されます。

nfeqn.data ファイルには、通常通り全エネルギーや原子に働く力の最大値のほか、ストレステンソルの最大値 (**sw_uniform** が on の場合は対角要素の平均値) が記録されます。たとえば、以下のような出力が得られます。

```

iter_unitcell, iter_ion, iter_total, etotal, forcex, stressmx
  1      1      18    -181.4043211413      0.0020128619
  1      2      27    -181.4043355689      0.0015666906
  1      3      36    -181.4043464493      0.0011267018
  1      4      44    -181.4043509953      0.0008837770
  1      5      53    -181.4043582176      0.0000137026      0.0002326236
  2      1      73    -181.4044226903      0.0000645338      0.0002272841
.....
.....

```

nfdynm.data ファイルも通常のものとはほぼ同じですが、通常の計算の場合は一度しか出力されないヘッダーが、セルベクトルが変形される度に出力されます。

```

#
#  a_vector =      8.6795114819      0.0000000000      0.0000000000
#  b_vector =      0.0000000000      8.6795114819      0.0000000000
#  c_vector =      0.0000000000      0.0000000000      5.5916992108
#  ntyp =      2 natm =      6
# (natm->type)      2      2      1      1      1      1
# (speciesname)      1 :      O
# (speciesname)      2 :      Ti
#
cps and forc at (iter_ion, iter_total =      1      18 )
  1      0.000000000      0.000000000      0.000000000      0.000000      0.000000      0.000000
  2      4.339755741      4.339755741      2.795849605      0.000000      0.000000      0.000000
  3      2.643779197      2.643779197      0.000000000      -0.001423      -0.001423      0.000000
  4      6.983534938      1.695976544      2.795849605      -0.001423      0.001423      0.000000
.....
.....
#
#  a_vector =      8.7672856463      0.0000000000      0.0000000000
#  b_vector =      0.0000000000      8.7672856463      0.0000000000
#  c_vector =      0.0000000000      0.0000000000      5.6429940606
#  ntyp =      2 natm =      6
# (natm->type)      2      2      1      1      1      1
# (speciesname)      1 :      O
# (speciesname)      2 :      Ti
#
cps and forc at (iter_ion, iter_total =      1      111 )
  1      0.000000000      0.000000000      0.000000000      0.000000      0.000000      0.000000
  2      4.383642823      4.383642823      2.821497030      0.000000      0.000000      0.000000
  3      2.663907294      2.663907294      0.000000000      0.001773      0.001773      0.000000
  4      7.047550117      1.719735530      2.821497030      0.001773      -0.001773      0.000000
  5      1.719735530      7.047550117      2.821497030      -0.001773      0.001773      0.000000

```

6	-2.663907294	-2.663907294	0.000000000	-0.001773	-0.001773	0.000000
					
					

5.6.1.3 計算例：ルチル型 TiO₂

入力パラメータファイルには、以下のような設定を施しました。

- カットオフエネルギーは 80 Rydberg
- 擬ポテンシャルはポータルサイトにおいて公開されている Ti_ggapbe_us_02.pp と O_ggapbe_us_02.pp
- 原子座標の最適化を施す設定；手法は BFGS 法，収束判定となる力の最大値は 2e-4
- 初期原子配置および格子定数は，無機材料データベース AtomWork (<http://crystdb.nims.go.jp/>) に登録されていたルチル型 TiO₂ のデータを採用
- 波動関数ソルバー，電荷密度ミキサーは指定せず，デフォルト設定を採用。

採用したカットオフエネルギーは 80 Rydberg と比較的大きなものですが，後述のように TiO₂ の場合はこれくらい必要であると考えられます。

nfefn.data ファイルの内容は，以下のようになりました。

iter	unitcell	iter_ion	iter_total	etotal	forcmx	stressmx
1	1	18	-181.4043211413	0.0020128619		
1	2	27	-181.4043355689	0.0015666906		
1	3	36	-181.4043464493	0.0011267018		
1	4	44	-181.4043509953	0.0008837770		
1	5	53	-181.4043582176	0.0000137026		0.0002326236
2	1	73	-181.4044226903	0.0000645338		0.0002272841
3	1	92	-181.4044839579	0.0001241955		0.0002222588
4	1	111	-181.4056948858	0.0025074070		0.0002222588
4	2	120	-181.4057176163	0.0020195652		0.0002222588
4	3	130	-181.4057600852	0.0000156213		0.0000444895
					
					
9	1	248	-181.4058191217	0.0001647915		0.0000332105
10	1	268	-181.4058328662	0.0000709369		0.0000119789
11	1	287	-181.4058349707	0.0000268520		0.0000015502
12	1	306	-181.4058351835	0.0000244918		0.0000006790

まずは，原子座標の最適化が 5 回実施されています。その間ストレステンソルは未計算なので，6 列目は空欄になっています。5 回目で原子に働く力の最大値が閾値より小さくなったので，セルを変形させたのちに計算が進行しています。この際に，単位胞最適化の更新回数を表す 1 列目の数値が 2 になっていることがわかります。また，6 列目にストレステンソルの最大値が記録されています。2 回目と 3 回目の更新時はセルを変形させても原子に働く力の最大値は閾値以下だったので原子座標の最適化は実施されませんでした，4 回目セルベクトル更新時にはそうではなかったので原子座標の最適化が行われています。このようにセルの最適化と必要に応じた原子座標の最適化が行われつつ計算が進行し，セルの更新回数が 12 回となったところでストレステンソルの最大値が閾値以下となったので計算は収束したとみなされ終了しています。単位胞最適化収束の履歴を，図にまとめました。

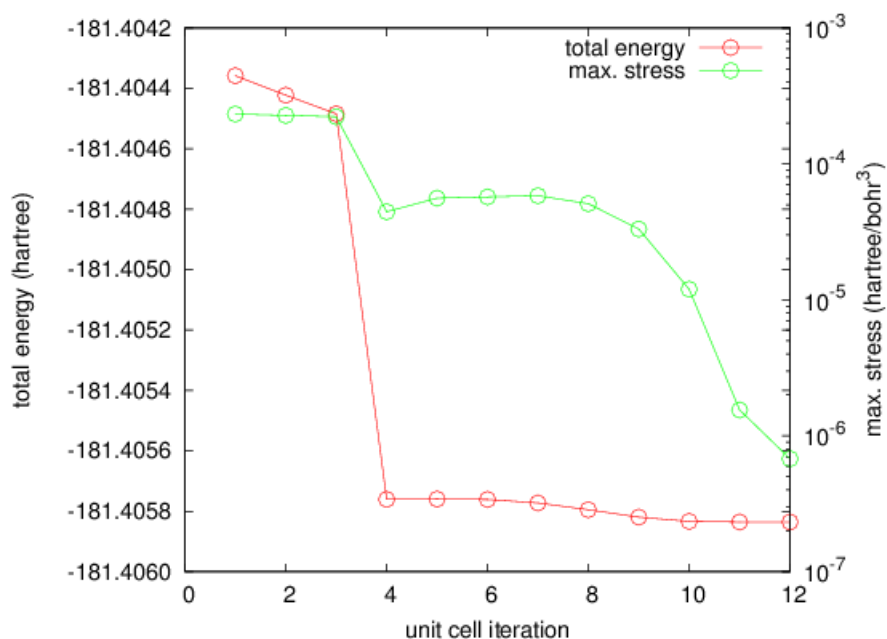


図 5-65 単位胞最適化の履歴。赤線：全エネルギー，緑線：ストレステンソルの最大成分。

安定な格子定数は、`nfdynm.data` ファイルに記録された最後のセルベクトル更新の情報からとめることができます。この例の場合、 $a=8.7934 \text{ bohr}$, $c=5.6164 \text{ bohr}$ と得られました。

● ストレステンソルとカットオフエネルギー

ストレステンソルは、全エネルギーや原子間力と比較してカットオフエネルギーに対して収束しづらい傾向があります。例として、ルチル型 TiO_2 の、実測値の格子定数で計算したストレステンソルとカットオフエネルギーの関係を図にプロットしました。

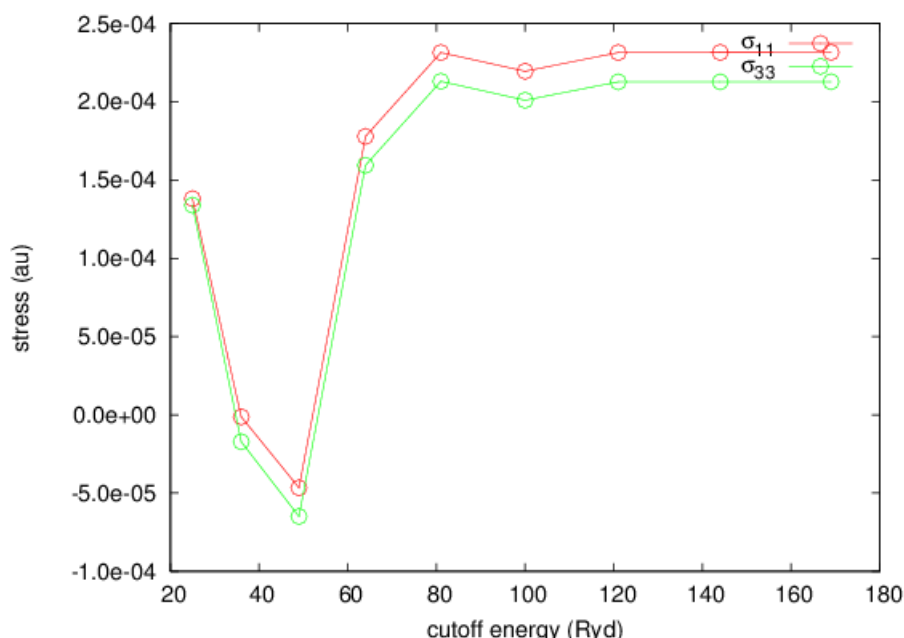


図 5-66 ルチル型 TiO_2 の場合の、ストレステンソルとカットオフエネルギーの関係

図からわかるように、カットオフ 50 Rydberg 程度の場合ストレステンソルの符号が間違っています。このケースでは、ある程度収束したストレステンソルを得るためには、最低でも 80 Rydberg 程度以上のカットオフエネルギーが必要であることが示唆されます。この状況は、5.1.1.4 で説明した補正によって改善するので、その点も考慮して採用するカットオフエネルギーを決めるようにしてください。

5.7 ノンコリニア系の計算、スピン軌道相互作用計算

5.7.1 ノンコリニア系の計算

5.7.1.1 機能の概要

通常の計算では、局所的な磁気モーメントの大きさは、アップ・ダウンスピン電子密度の差で表現され、その向きについては規定されません。向きまで表現するためには、1つの波動関数がアップ及びダウンスピン成分をもつようにする、すなわち2成分スピノールで表す必要があります。

$$\begin{pmatrix} \psi_{nk}^{\uparrow} \\ \psi_{nk}^{\downarrow} \end{pmatrix} = \begin{pmatrix} \psi_{nk}^{\uparrow}(G=0) \\ \vdots \\ \psi_{nk}^{\uparrow}(G=G_{\max}) \\ \psi_{nk}^{\downarrow}(G=0) \\ \vdots \\ \psi_{nk}^{\downarrow}(G=G_{\max}) \end{pmatrix}$$

これに対応して電荷密度は

$$n^{\alpha\beta}(\mathbf{r}) = \sum_{nk} f_{nk} \langle \psi_{nk}^{\alpha} | \mathbf{r} \rangle \langle \mathbf{r} | \psi_{nk}^{\beta} \rangle$$

のように、スピン指標に関して2x2行列となります。また、局所的な電荷密度及び磁気モーメントは、

$$n_{\text{tot}}(\mathbf{r}) = \text{Tr}[n(\mathbf{r})] = n^{\alpha\alpha}(\mathbf{r}) + n^{\beta\beta}(\mathbf{r})$$

$$m_x(\mathbf{r}) = n^{\alpha\beta}(\mathbf{r}) + n^{\beta\alpha}(\mathbf{r})$$

$$m_y(\mathbf{r}) = i[-n^{\alpha\beta}(\mathbf{r}) + n^{\beta\alpha}(\mathbf{r})]$$

$$m_z(\mathbf{r}) = n^{\alpha\alpha}(\mathbf{r}) - n^{\beta\beta}(\mathbf{r})$$

となります。

5.7.1.2 入力パラメータ

ノンコリニア系の計算、すなわち2成分スピノールでの計算を行うには、structureブロックに“magnetic_state = noncollinear”と記します。また、各原子種の局所磁気モーメントの方向の初期値を、“mdx mdy mdz”に指定します。特に指定しない場合は、z方向を向くと判断します。局所磁気モーメントの方向の初期値は、“theta phi”(単位: degree)でも指定可能です。

```
structure{
    ...
    magnetic_state = noncollinear

    element_list{
        #units atomic_mass
        #tag element atomicnumber    zeta    deviation    mdx    mdy    mdz
        0      8      0.166666    1.5    0.0    0.0    1.0
    }
}
```

```

    }
    ...
}

```

5.7.1.3 計算結果の出力

標準出力には、以下のような磁気モーメントに関する情報が出力されます。**Tot, Mx, My, Mz** は、それぞれ、局所的な電荷密度及び磁気モーメントを単位胞内で足し合わせた値です。これ以外は通常の計算による出力と共通です。

```

!OLD Chg ** Tot: 40.00000000 Mx: 0.00073742 My: 0.00000000 Mz: 16.17742289
!NEW Chg ** Tot: 40.00000000 Mx: 0.00075559 My: 0.00000000 Mz: 16.17739619

```

5.7.2 スピン軌道相互作用計算

5.7.2.1 機能の概要

2成分スピノールが重要になるのは、スピン軌道相互作用を考慮する場合です。スピン軌道相互作用は、

$$H_{\text{soc}} = \frac{\hbar^2}{2m^2c^2} \frac{1}{r} \frac{\partial V_c(r)}{\partial r} \mathbf{L} \cdot \mathbf{S}$$

で表されます。ここで、 $V_c(r)$ は原子核周りの球対称なポテンシャルです。この **Hamiltonian** は、波動関数のアップ及びダウンスピン成分の間に相互作用を働かせるため、5.8.1 で説明した 2 成分スピノールが必要になります。

5.7.2.2 入力パラメータ

スピン軌道相互作用を利用するには、**accuracy** ブロック内の **spin_orbit** ブロックに “**mode = pawpot** “ と記します。これ以外は、ノンコリニア系の計算と同様です。

```

accuracy{
  ...
  spinorbit{
    mode = pawpot
  }
  ...
}

```

5.7.2.3 計算結果の出力

スピン軌道相互作用によるエネルギーは、標準出力に **ESpinOrb_old, new** で表示されます。

なお、これらの値は、コンパイル時に **CPPFLAG** に **-DUSE_ESPINORB** をつけた場合にのみ計算されます。

```

TOTAL ENERGY FOR 53 -TH ITER= -41.454944288742 edel = -0.170628D-08 : SOLVER = SU
BMAT + RMM3
KI= 13.204535394898 HA= 32.283599969986 XC= -6.801519951682 LO= -83.580054495015
NL= 7.597059454569 EW= 5.402894293900 PC= 0.000000000000 EN= 0.000000000000
PHYSICALLY CORRECT ENERGY = -41.454944288742
EOHXC_PAW= -0.4786729 HA_PAW= 0.0218350
XC_PAW_AE= -15.6619733 XC_PAW_PS= -5.6000049
!XC_PAW_AE-XC_PAW_PS= -10.0619684
ESpinOrb_old= -0.0000308 ESpinOrb_now= -0.0000293

```


5.7.2.4 計算例：O₂分子、Pt₂分子

計算例は、sample/SpinOrbit です。これらの例では、分子を x 軸方向に配置し、磁気モーメントの向き (theta) による全エネルギーの違いを計算します。

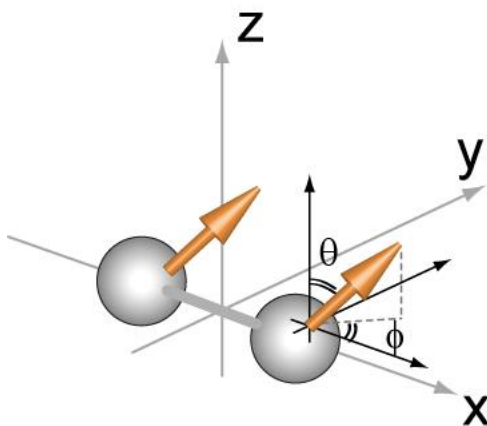


図 5-67 スピン軌道相互作用計算の計算例の分子配置

O₂分子

- SpinOrbit/O2/Theta_0
- SpinOrbit/O2/Theta_90

前者の方が、1原子あたり 0.108 meV 安定です。

Pt₂分子

- SpinOrbit/Pt2/Theta_0
- SpinOrbit/Pt2/Theta_90

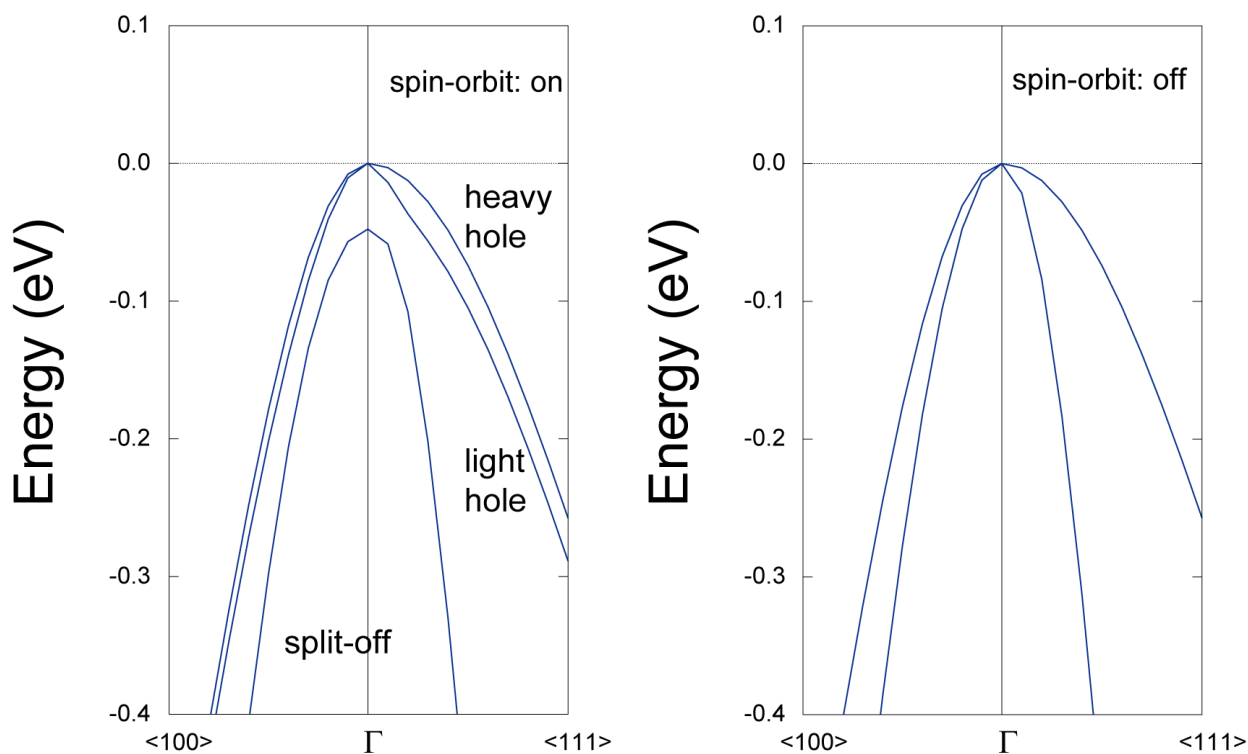
後者の方が、1原子あたり 17.215 meV 安定です。

5.7.2.5 Si 結晶のバンド

スピン軌道相互作用を考慮したバンド計算は、通常のコリニア計算と同じ手順で行います。まず、scf 計算を行い (phase)、得られた電荷密度のもとでバンド計算 (ekcal) を行います。

例として、Si diamond 結晶のバンド計算を示します。scf 及び band 計算ともに、以下のキーワードを用います (Si-band/soc-on フォルダ参照)。

```
accuracy{
  paw = on
  spinorbit{
    mode = pawpot
  }
}
structure{
  magnetic_state = noncollinear
}
```



左図は上記入力により得られたバンド構造で、右図はスピン軌道相互作用を考慮しないバンド構造です。スピン軌道相互作用により Γ 点の縮退が解けていることが分かります。

5.7.2.6 使用上の注意

- スピン軌道相互作用を用いてエネルギー比較を行う場合、対称操作を **E** のみにされることをお勧めします。
- smearing において method = parabolic のみを使用可能です。

5.8 ELNES / XANES 解析機能

5.8.1 はじめに

EELS は電子が試料を透過する際のエネルギー損失を測定し、XAFS は試料に X 線を照射した際に現れるスペクトルを測定する実験手法です。ここでは、電子線あるいは X 線により、内殻電子が非占有状態に遷移する現象を対象とします。また、広範なエネルギー領域の中で、特に、吸収端付近のスペクトル構造を計算するものとします。この狭いエネルギー領域では、EELS は ELNES、XAFS は XANES と呼ばれています。

ELNES と XANES の二重微分散乱断面積は、双極子近似のもとで、以下のように表されます。

$$\begin{aligned}\frac{\partial^2 \sigma}{\partial E \partial \Omega} &\propto \sum_{i,f} \left| \langle i | \mathbf{q} \cdot \mathbf{R} | f \rangle \right|^2 \quad (\text{ELNES}), \\ \frac{\partial^2 \sigma}{\partial E \partial \Omega} &\propto \sum_{i,f} \left| \langle i | \mathbf{e} \cdot \mathbf{R} | f \rangle \right|^2 \quad (\text{XANES})\end{aligned}\tag{1}$$

ここで $|i\rangle, |f\rangle$ は始状態（基底状態）と終状態（ある原子の内殻電子が飛び出し、非占有状態に遷移した状態）、 \mathbf{q} は非弾性散乱時における波数の変化、 \mathbf{e} は入射 X 線の偏光ベクトルである。したがって、XANES の計算は、ELNES 計算にて \mathbf{q} を \mathbf{e} と読み替えます。内殻軌道 $|\phi_c\rangle$ から状態 n, k で指定される伝導バンド $|\psi_{nk}\rangle$ への遷移確率は、文献[1] に従い、

$$\langle \phi_c | r_\alpha | \psi_{nk} \rangle = \langle \phi_c | r_\alpha | \tilde{\psi}_{nk} \rangle + \sum_i \left(\langle \phi_c | r_\alpha | \phi_i \rangle - \langle \phi_c | r_\alpha | \tilde{\phi}_i \rangle \right) \langle \tilde{p}_i | \tilde{\psi}_{nk} \rangle.\tag{2}$$

で与えられるものとします。ここで 右辺第 1 項は、波動関数の soft part の寄与、第 2 項は擬ポテンシャルを用いたことによる補正項です。 $|\phi_i\rangle$ は原子軌道 i の全電子波動関数、 $|\tilde{\phi}_i\rangle$ は原子軌道 i の擬波動関数である。

また、吸収端のエネルギーは、基底状態と内殻電子励起状態の全エネルギー差で与えられるものとします。

5.8.2 準備：擬ポテンシャル

公開されている擬ポテンシャルには、内殻軌道に関する情報が含まれていません。このため、ELNES・XANES の計算を行うためには、擬ポテンシャルをあらためて作成する必要があります。作成すべき擬ポテンシャルは、

- 基底状態の擬ポテンシャル
- 内殻に空孔を入れた状態の擬ポテンシャル

です。例えば、LiF 結晶の Li 原子の K 端のスペクトルを計算する場合には、

- Li 原子の基底状態の擬ポテンシャル
- Li 原子の 1s 軌道に空孔を入れた擬ポテンシャル
- F 原子の基底状態の擬ポテンシャル

を作成します。なお、いずれの擬ポテンシャル作成においても、CIAO の入力ファイルに、

<code>sw_write_core_to_valence 1</code>

を記入する必要があります。キーワードについては CIAO のマニュアルを参照。

5.8.3 計算の流れ

計算は以下の順で行います。

- ① 基底状態 (A) の SCF 計算
- ② 内殻電子励起状態 (B) の SCF 計算
- ③ 内殻電子励起状態 (B) のスペクトル計算

①と②で得られた(A)と(B)の全エネルギー差をスペクトルの吸収端エネルギーとし、③の計算に反映します。

なお、実行コマンドは、①及び②の SCF 計算は `phase`、③のスペクトル計算は `epsmain` を使用します。

5.8.4 基底状態 (A) の SCF 計算

5.8.4.1 入力

吸収端のエネルギーを計算するために、入力パラメータに以下のキーワードを設定します。

```
accuracy{
  paw = on
}
Postprocessing{
  CoreLevels{
    sw_calc_core_energy = on
  }
}
```

なお、file_names.data の F_POT に、5.8.2 で作成した擬ポテンシャルファイル名を指定します。通常の、内殻軌道情報を含まない擬ポテンシャルを指定した場合には、該当原子の内殻電子の寄与を考慮せずにエネルギー計算が行われます。

```
&fnames
  F_INP    = './nfinp1.data'
  F_POT(1) = './pp/Li_ggapbe_paw_02_no_corehole.pp'
  F_POT(2) = './pp/F_ggapbe_paw_02_no_corehole.pp'
&end
```

5.8.4.2 出力

core_energy.data の最終行に、内殻電子と価電子の寄与を合わせた全エネルギー値が出力されます。core_energy.data 以外のファイルに出力したい場合には、file_names.data で F_CORE_ENERGY_OUT にファイル名を指定します。

```
# Etotal (Core+Valence)
-3437.92292869603
```

5.8.5 内殻電子励起状態 (B) の SCF 計算

5.8.5.1 入力

基底状態の計算と同じキーワードを用います。その他の注意事項として、内殻電子を励起する原子には、別の原子種を割り当てる必要があります。また、励起原子は単位胞に 1 原子のみとする。以下の例では、励起原子は Li2 です。

```
structure{
  atom_list{
    atoms{
      #tag element rx ry rz mobile
      Li2      0.0000000000    0.0000000000    0.0000000000
      F1       0.2500000000    0.0000000000    0.0000000000
      ...
      (中略)
      ...
      Li1      0.7500000000    0.5000000000    0.7500000000
      F1       1.0000000000    0.5000000000    0.7500000000
    }
  }
  element_list{
```

```

#units atomic_mass
#tag element atomicnumber zeta deviation
  Li1      3      0.00  1.5
  Li2      3      0.00  1.5
  F1       9      0.00  1.5
}
}

```

Li2 には、内殻電子を励起させて作成した擬ポテンシャルを割り当てる。内殻電子励起状態 (B) の SCF 計算の `file_names.data` の記述例です。

```

&fnames
  F_INP      = './nfinp1.data'
  F_POT(1) = './pp/Li_ggapbe_paw_02_no_corehole.pp'
  F_POT(2) = './pp/Li_ggapbe_paw_02_1s_corehole.pp'
  F_POT(3) = './pp/F_ggapbe_paw_02_no_corehole.pp'
&end

```

5.8.5.2 出力

基底状態の計算と同様に、`core_energy.data` の最終行に、内殻電子と価電子の寄与を合わせた全エネルギー値が出力されます。

5.8.6 内殻電子励起状態 (B) のスペクトル計算

5.8.6.1 入力

スペクトル計算の入力では、`control` ブロックにて `condition = fixed_charge` を指定します。

```

control{
  condition = fixed_charge
  use_additional_projector=on
}

```

また、`structure` ブロックは、基底状態の計算と同様に記述します。

`epsilon` ブロックでは、スペクトルに関するパラメータ指定を行う。`sw_corelevel_spectrum` は、内殻軌道からの励起スペクトル計算を行うためのスイッチで、`on` と指定します。`probe` ブロック内の `atom_id` には、励起させる原子の番号を指定します。上記の `Li2` は1番目の原子であるため、`atom_id = 1` としています。また、`orbital` には、擬ポテンシャル作成時に空孔を入れた軌道を指定します。この例では `orbital` に `1s` を指定しており、Li K 端におけるスペクトルが計算されます。

`fermi_energy` ブロックで `read_efermi = on` とし、`efermi` には、内殻電子励起状態 (B) の SCF 計算の `nfefermi.data` 記載の値を指定します。

`energy` ブロックには、吸収端を基準にして観察したいエネルギー領域を指定する。`low`, `high`, `step` は、それぞれ、エネルギー領域の下端、上端、及び間隔に対応する。単位は `hartree` です。

XANES の計算の入力パラメータは、EELS の際は `photon` を `eels` と書き換えます。なお、内部の処理は共通であるため、どちらのキーワードを用いても結果は変わりません。

```

epsilon {
  sw_epsilon = on
  sw_corelevel_spectrum = on
  probe{

```

```

        atom_id = 1
        orbital = 1s
    }
    fermi_energy{
        read_efermi = on
        efermi = 0.21930399
    }
    photon{
        energy{
            low = -0.10, high = 2.0, step = 0.002
        }
    }
    transition_moment{
        type = ks ! {l | rn | ks}
        symmetry = on
    }
    BZ_integration {
        method = g ! {parabolic(p) | gaussian(g) | tetrahedron(t)}
        width = 1.0 eV
    }
}

```

最後に、file_names.data では、基底状態 (A) 及び内殻電子励起状態 (B) の全エネルギー値出力ファイル名を、F_CORE_ENERGY_INITIAL 及び F_CORE_ENERGY_FINAL に指定します。また、F_CHGT 及び F_CNTN_BIN_PAW に、内殻電子励起状態 (B) の電荷密度ファイルを指定します。

```

&fnames
F_INP    = './nfinp1.data'
F_POT(1) = './pp/Li_ggapbe_paw_02_no_corehole.pp'
F_POT(2) = './pp/Li_ggapbe_paw_02_1s_corehole.pp'
F_POT(3) = './pp/F_ggapbe_paw_02_no_corehole.pp'

F_CHGT   = './scf_excited/nfchgt.data'
F_CNTN_BIN_PAW = './scf_excited/continue_bin_paw.data'

F_CORE_ENERGY_INITIAL = './scf_ground/core_energy.data'
F_CORE_ENERGY_FINAL   = './scf_excited/core_energy.data'
&end

```

5.8.6.2 出力

スペクトルの出力ファイル名は eps.data である。以下に出力例を示す。3 行目のエネルギー値は、吸収端エネルギーに e_low を加えた値です。spectrum はスペクトル強度、すなわち誘電関数の虚部 $\epsilon_2(\omega)$ に対応します。

```

#           Spectrum data
#   Energy[eV]      Spectrum
0.6018296869E+02  0.7167942086E-06
0.6023739146E+02  0.1031555487E-05
0.6029181423E+02  0.1478755242E-05
...

```

5.8.7 計算事例

5.8.7.1 LiF 結晶の Li 原子 K 端のスペクトル

例題として、LiF 結晶の Li 原子 K 端の XANES スペクトルを示します。計算事例は、samples/XANES/LiF です。計算は、

- ① 基底状態 (A) の SCF 計算 (フォルダ名: scf_ground)
- ② 内殻電子励起状態 (B) の SCF 計算 (フォルダ名: scf_excited)
- ③ 内殻電子励起状態 (B) のスペクトル計算 (フォルダ名: eps_excited)

の順で行います。

スペクトル計算は k 点数を増やして行うことを推奨します。

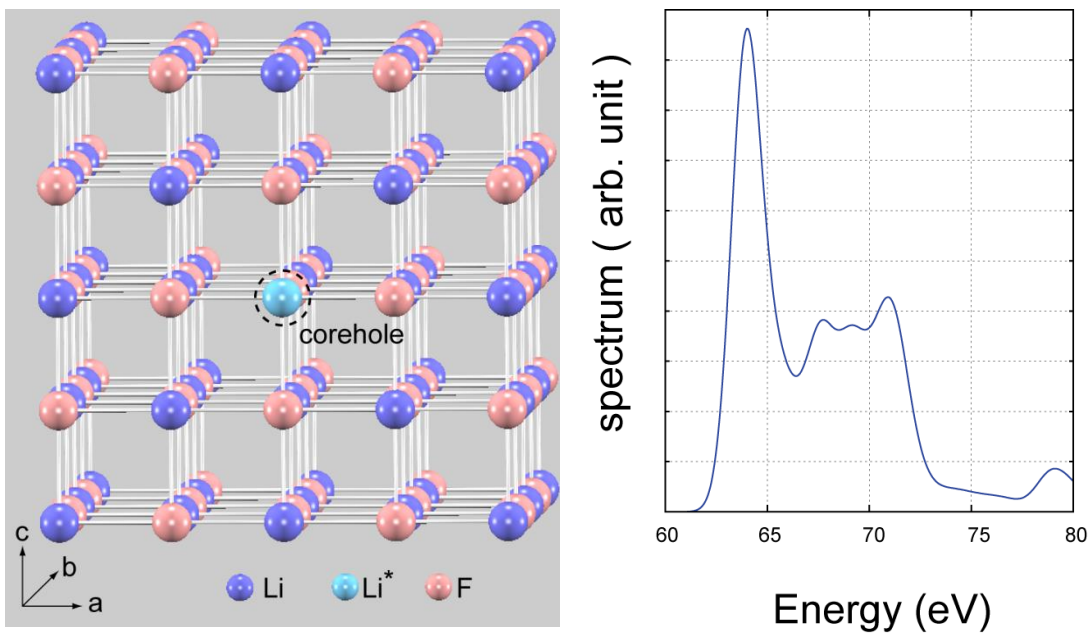


図 5-68

図 5-69 (左) LiF 結晶。(右)スペクトル

5.8.8 文献

[1] S. -P. Gao, C. J. Pickard, M. C. Payne, J. Zhu, and J. Yuan, Phys. Rev. B **77** (2008) 115122.

5.9 化学ポテンシャル一定のシミュレーション

5.9.1 概要

PHASE/0 による通常のシミュレーションでは、電子数一定の計算を行います。これに対し、化学ポテンシャル（フェルミエネルギー）を一定とし、構造最適化や分子動力学シミュレーション、NEB 計算などを行う[1]ことができる機能（constant-mu 法）も PHASE/0 には備わっています。この場合、電子数はシミュレーション中変化します。

5.9.2 入力パラメーター

化学ポテンシャル一定のシミュレーションを行うには、accuracy ブロックの下に fcp ブロックを作成し、次の例のように設定を行います。

```
accuracy{  
  ...  
  fcp{  
    sw_fcp = ON  
    mu = -0.1  
    relax_crit = 1.0d-5  
  }  
  ...  
}
```

fcp ブロックにおいて定義可能なパラメータは下記の通りです。

sw_fcp	化学ポテンシャル一定の計算を行うかどうかを指定するスイッチです。行う場合に on とします。
mu	ターゲットとなる化学ポテンシャルの値をエネルギーの単位で指定します。事前に通常の計算で状態密度を計算しておき、フェルミエネルギー近辺のエネルギー固有値の分布や電子数などを調べておく設定しやすいでしょう。
temperature	化学ポテンシャルを制御するための“粒子浴”の温度を指定します。分子動力学シミュレーションの場合に意味のある設定です。
qmass	化学ポテンシャルを制御するための“粒子浴”の質量を指定します。分子動力学シミュレーションの場合に意味のある設定です。
mass	化学ポテンシャルを制御するための“電子の質量”を指定します。分子動力学シミュレーションの場合に意味のある設定です。
relax_crit	構造最適化の際の収束判定条件を指定します。得られる化学ポテンシャルの値とターゲットの値の差の絶対値がここで指定する値よりも小さくなった場合に収束したと見なされます（これ以外に、通常の原子間力に関する収束判定条件も考慮されます）。
tot_charge_first	NEB 計算の際に、始点のレプリカに与える電荷を指定します。
tot_charge_last	NEB 計算の際に、終点のレプリカに与える電荷を指定します。

なお、本計算機能を使って（ジョブ 1 とする）得られた（中性でない）電荷を外部電荷としてあたえて通常の計算を行う（ジョブ 2 とする）場合、（ジョブ 1 において）与えた mu の値とは異なるフェルミエネルギーが（ジョブ 2 において）得られます。逆に、外部電荷を与えて通常の計算を行って（ジョブ 3 とする）得られたフェルミエネルギーを mu に指定して（ジョブ 4 によって）最適化を行うと、得られる電荷はもとの（ジョブ 3 における）外部電荷とは異なる電荷となります。このようにつじつまの合わない結果が得られるのは、中性でない電荷を与える場合は初期電荷の与え方が中性の場合と異なるので、エネルギーの原点が変化するためです。つじつまの合った結果を得るためには、外部電荷を与えた計算について以下のような設定を加え、初期電荷の計算方法を合わせるようにしてください。

```
accuracy{
    ...
    sw_add_qex_to_initial_charge = off
    ...
}
```

5.9.3 計算の実行

通常の構造最適化もしくは分子動力学シミュレーションの設定に加え，化学ポテンシャル一定のシミュレーションの設定を施したら，通常通り PHASE/0 を実行すれば計算を行うことができます。

計算中に電荷がどのように変化したかは，以下の要領で調べることができます。

```
% grep 'Total Charge' output000
FCP : Total Charge =      31.98942095
FCP : Total Charge =      31.99795170
FCP : Total Charge =      32.01254363
FCP : Total Charge =      32.02605805
FCP : Total Charge =      32.03237025
FCP : Total Charge =      32.02886985
FCP : Total Charge =      32.01742419
FCP : Total Charge =      32.00372842
FCP : Total Charge =      31.99503316
...
...
```

また，化学ポテンシャル（フェルミエネルギー）がどのように変化したかは，以下の要領で調べることができます。

```
% grep 'Fermi Energy' output002
FCP : Fermi Energy =      0.24621583
FCP : Fermi Energy =      0.24679898
FCP : Fermi Energy =      0.24700415
FCP : Fermi Energy =      0.24674191
FCP : Fermi Energy =      0.24618985
...
...
```

5.9.4 例題

非常に単純な例題として，シリコン 8 原子の系の分子動力学シミュレーションを取り上げます。この例題の入力ファイルは，samples ディレクトリーの FCP/NVT_nose_hoover にあります。

この例題の入力ファイルは，基本的には samples ディレクトリーの molecular_dynamics/NVT 以下にあるものと同等ですが，以下のように“化学ポテンシャル一定の分子動力学シミュレーション”を行うための設定が施されています。

```
accuracy{
    ...
    fcp{
        sw_fcptopt = ON
        mu = 9.0e-3
    }
}
```

```

    mass = 1000.0d0
    qmass = 4000
  }
  ...
}

```

この入力ファイルを利用して、通常通り **PHASE/0** を実行すれば化学ポテンシャル一定の第一原理分子動力学シミュレーションを行うことが可能です。その結果、たとえば図 5-70 に示すように電荷がシミュレーション中時々刻々と変化します。

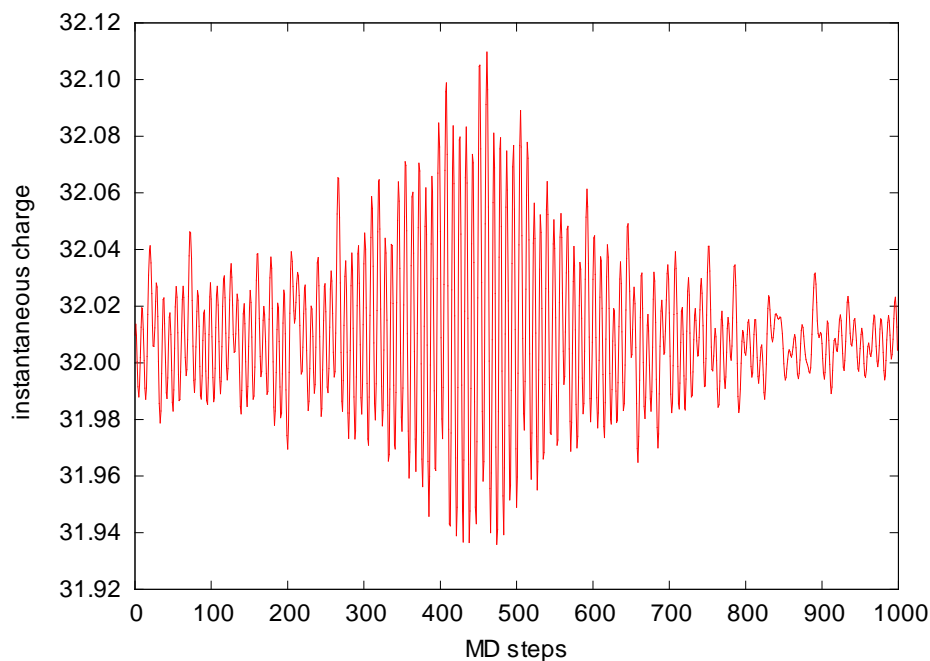


図 5-70 分子動力学シミュレーション中の電荷の変化。

5.9.5 使用上の注意

本計算機能は、すべての機能と組み合わせて利用することが可能ですが、本計算機能を利用すると電荷中性ではない計算を行うことになる点には注意が必要です。有効遮蔽体法（ESM 法。5.3.5 参照）で境界条件を **pe1**（両側の ESM が金属）とすると、隣り合う単位胞の相互作用の影響を取り除くこともできます。

5.9.6 参考文献

- [1] N. Bonnet, T. Morishita, O. Sugino, and M. Otani, “First-Principles Molecular Dynamics at a Constant Electrode Potential”, *Physical Review Letters* **109** 266101 (2012).

6. PAW 法による計算

6.1 機能の概要

PAW 法とは, projector-augmented wave 法の略称です。ウルトラソフト擬ポテンシャル法と深い関係のある計算手法ですが, ウルトラソフト擬ポテンシャル法と比較すると, 特に磁性を考慮する場合や電荷移動の大きな系において高精度な計算が行えるとされている計算手法です。ここでは, PAW 法による計算を PHASE で実行する方法を説明します。

6.2 入力パラメータ、計算の実行方法

PAW 法を利用するためには, PAW ポテンシャルを利用する必要があります。PAW ポテンシャルは, 擬ポテンシャル格納ディレクトリーに

```
元素名_ggapbe_paw_us.pp
元素名_ggapbe_paw_nc.pp
```

というファイル名で存在します。通常のノルム保存およびウルトラソフトポテンシャルと同様, file_names.data ファイルにおいて F_POT 識別子で対応する元素名のファイルを指定します。さらに, 以下のように paw を利用するような指定を入力ファイルに記述します。

PAW ポテンシャルを指定しても, デフォルトの状態では PAW の計算は行われません。PAW 法の計算を行うには, 入力ファイルの accuracy ブロックで変数 paw を定義し, その値を on にする必要があります。

```
accuracy{
    paw = on
}
```

PAW 法の場合, 欠損電荷の扱い方を変更することによって収束性を向上させることができる場合があります。以下の設定を 施すことにより, 多くの場合収束が加速されます。

```
charge_mixing{
    ...
    sw_mix_charge_hardpart = on
}
```

ekcal による固定電荷の計算を行う場合, さらに file_names.data に F_CNTN_BIN_PAW 識別子を利用して PAW 計算用のデータを指定し, 読み込ませる必要があります。このファイルの既定のファイル名は continue_bin_paw.data です。たとえば SCF 計算を行ったディレクトリが 1 階層上のディレクトリだった場合, 下記のような記述が必要です。

```
&fnames
...
...
F_CNTN_BIN_PAW='../continue_bin_paw.data'
/
```

6.3 計算例：体心立方構造クロム

PAW 法を利用した計算例として, 体心立方構造クロムの格子定数の計算例を紹介します。クロムは, ウルトラソフト擬ポテンシャルで計算すると格子定数が過大評価され, また体積弾性率が過小評価されます。この点がどのように改善されるか確認します。

計算に利用したデータは, samples/Cr にあります。samples/Cr には, 以下のファイルとディレクトリがあります。

Cr_ggapbe_paw_002.gncpp2

```
Cr_ggapbe_us_02.pp
paw/
us/
```

Cr_ggapbe_paw_002.gncpp2 が PAW 用のポテンシャルファイル、Cr_ggapbe_us_02.pp がウルトラソフト擬ポテンシャルのファイルです。ディレクトリ paw に PAW 用の入力データが、us にウルトラソフト用の入力データが格納されています。さらに paw, us の各ディレクトリにも次のファイルとディレクトリがあります。

```
catenergy.sh
vol20/
vol21/
.....
```

catenergy.sh は、すべての体積についての計算が終了したあとにエネルギーデータファイルを結合するシェルスクリプトです。vol20, vol21, ... はそれぞれ体積 20\AA^3 , 21\AA^3 , ... の入力データに対応します。すべてのディレクトリで計算を実行したあと、catenergy.sh を実行すると paw(us)ディレクトリ下に各体積のエネルギーを記録したファイル、energy.data が作成されます。

paw ディレクトリ下のディレクトリ (vol20 など) の中にある file_names.data ファイルには、以下のような記述があります。

```
F_POT(1) = '../..'/Cr_ggapbe_paw_002.gncpp2'
F_POT(2) = '../..'/Cr_ggapbe_paw_002.gncpp2'
```

この指定があるので、PAW 用のポテンシャルが利用されます。同じように、us ディレクトリ下のディレクトリ (vol20 など) の file_names.data ファイルには Cr_ggapbe_us_02.pp を指定する行があります。

計算の結果得られた EV 曲線を図 6-1 に示します。一見して明らかなように、PAW 法と US 法とでは異なる EV 曲線が得られます。さらに、この EV 曲線をもとに格子定数、体積弾性率、凝集エネルギーをもとめた結果を実測値とともに表 6.1 にまとめました。PAW 法は格子定数、体積弾性率が US 法よりも改善している (実験値との一致がよい) ことが分かります。

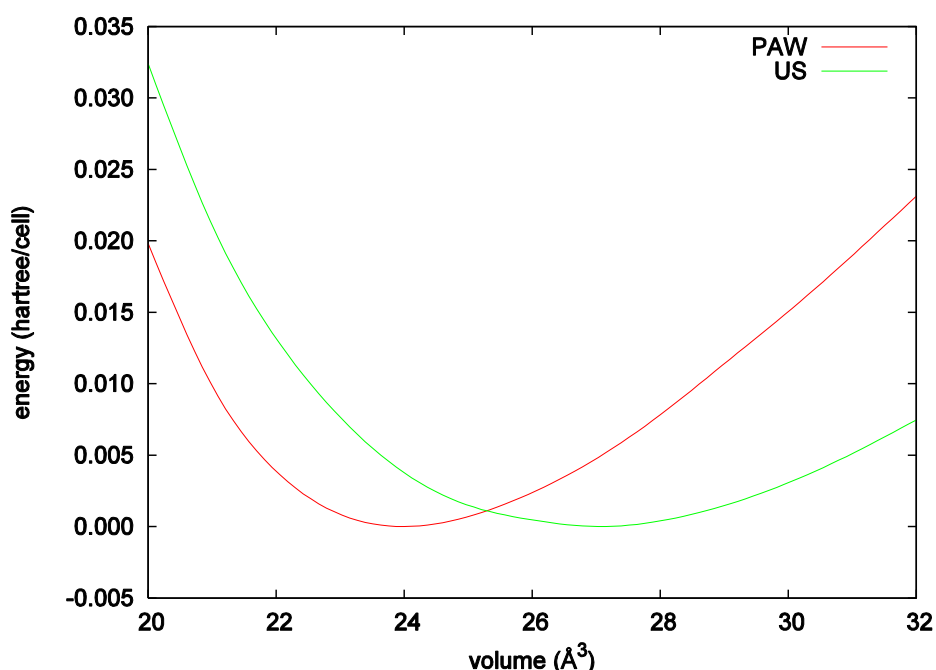


図 6-1 クロムの EV 曲線。赤線が PAW の結果、緑線が US の結果。各手法で得られた最も低いエネルギーをエネルギーの原点としている。

表 6.1 PAW 法および US 法によってもとめた格子定数, 体積弾性率, 凝集エネルギー

	US	PAW	実測
a (Å)	2.994	2.886	2.88
B (GPa)	89.2	150.5	190.1
E _{coh} (eV/atom)	4.01	3.065	4.10

6.4 PAW 法の精度を向上させる方法

PAW 法で実施される球面積分は, デフォルトでは球面調和関数を利用した近似による高速化がなされます。この積分を, 近似をせずに計算するオプションも用意されています。入力パラメーターファイルに以下のような記述を加えるとこのオプションを利用することができます。

```
paw_one_center_integral{
  element_list{
    #tag element surface_integral_method
    Fe gl
    O gl
  }
}
```

この例では, Fe と O という二種類の元素が定義されていることを仮定しました。利用する場合元素数分定義する必要があります。

この方法を使うと交換相関相互作用の処理時間が長くなってしまいますが, 精度が向上し (数値的により厳密に解けるようになり), SCF 計算の収束性が向上する場合があります。

6.5 PAW 法で有効な計算機能一覧

PAW 法で利用可能な機能です。

- 全エネルギー
- 対称性
- スピン分極
- 構造最適化
- 全電荷密度・部分電荷密度の出力
- 各種状態密度の計算
- バンド構造
- ストレステンソルの計算
- 仕事関数
- 振動解析
- 分子動力学
- DFT+U
- ESM 法
- 拘束条件付きダイナミクス
- メタダイナミクス
- NEB
- 単位胞最適化
- ノンコリニア磁性
- スピン軌道相互作用
- UVSOR-Epsilon の各機能

- UVSOR-Berry-Phonon の各機能

7. 計算精度など補足事項

7.1 計算精度、収束性

7.1.1 カットオフエネルギーと計算精度

平面波基底を採用している利点の1つとして、カットオフエネルギーは大きくすればするほど必ず全エネルギーは小さくなり、密度汎関数理論の厳密解に近づく、という点が挙げられます。具体例として、面心立方格子のアルミニウム結晶を利用したテスト例を紹介します。図 7-1 にカットオフエネルギーと全エネルギーの関係を示します。

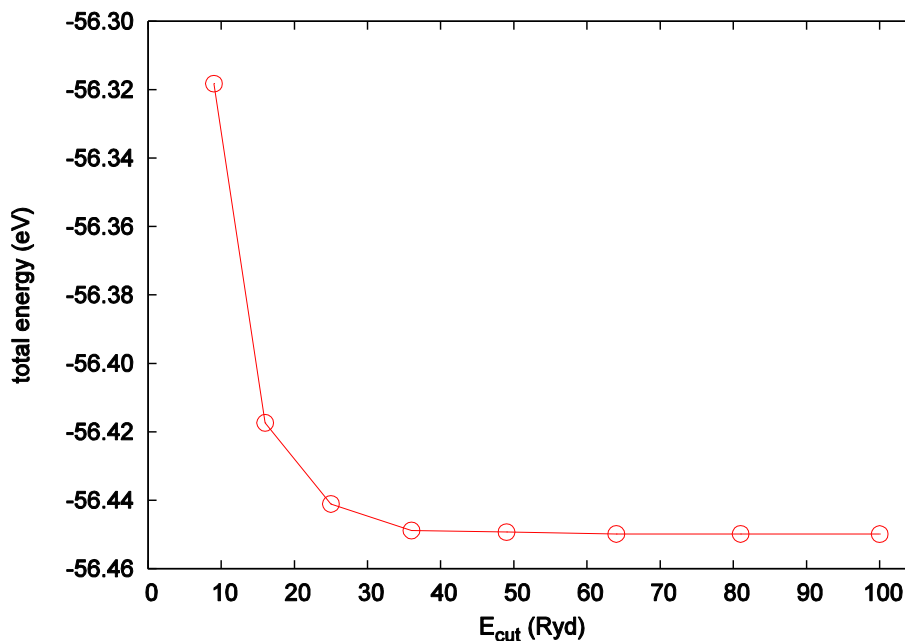


図 7-1 アルミニウム結晶の場合の、カットオフエネルギーと全エネルギーの関係

図から明らかなように、カットオフエネルギーを大きくすると全エネルギーが小さくなり、一定の値に収束しています。この振る舞いは利用している擬ポテンシャルに依存します。この例では、36 Rydberg ほどで原子あたり 1 meV 程度まで収束しています。どの程度の収束を目指すべきかは対象とする問題によって異なりますが、通常 10 meV 程度の収束が得られていれば十分であると考えられます。また、全エネルギーを絶対エネルギーで評価するのではなく、(ふたつの構造の全エネルギーの差など) 相対エネルギーで評価する場合はより小さなカットオフエネルギーで収束することが期待できます。

7.1.2 k 点サンプリングと計算精度

PHASE は平面波基底を採用しているので、扱える問題は周期系に限られます。したがって、すべての物理量は最終的には第一ブリュアンゾーン内で積分する必要があります。この第一ブリュアンゾーン内の積分の細かさを指定するのが k 点サンプリングです。 k 点サンプリング数と全エネルギーの関係を、図 7-2 に示します。

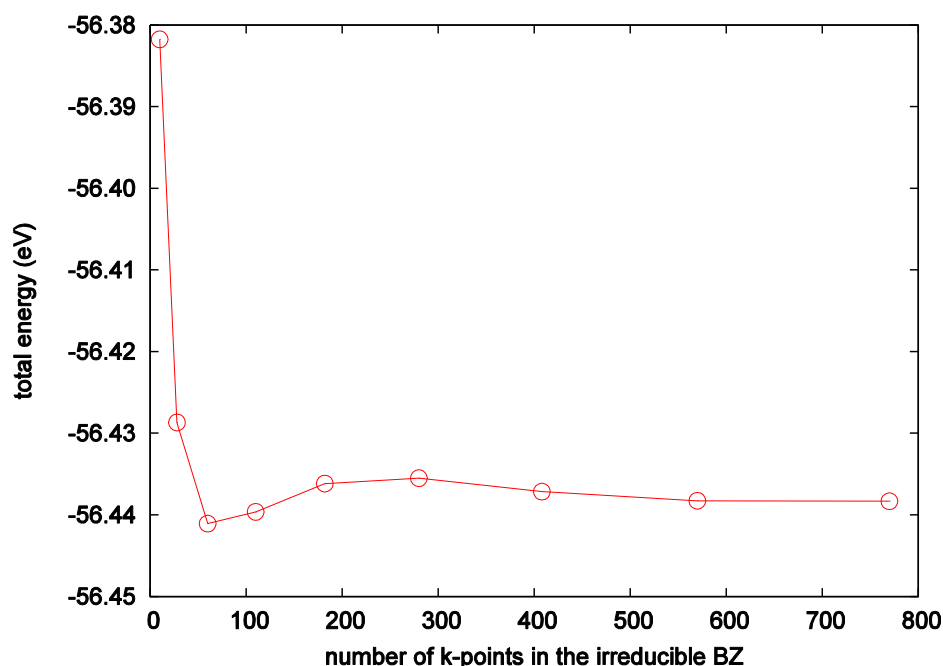


図 7-2 アルミニウム結晶の場合の、 k 点数と全エネルギーの関係

k 点数に関しては変分原理が成立するわけではないので、 k 点数に応じて全エネルギーが単純減少するわけではない点には 注意が必要です。図 30 の例でも、途中全エネルギーが大きくなってから収束へ至っていることが分かります。

なお、カットオフエネルギーの場合と同様ここでみた全エネルギーの絶対エネルギーではなく相対エネルギーの場合は より少ない k 点サンプリング数で収束することが期待できます。

7.1.3 収束判定と計算精度

SCF 計算の収束判定を厳しくすると、原子に働く力をより精度よく計算することが可能となります。通常の構造最適化の場合 10^{-8} hartree 程度の収束判定を採用すれば 多くの場合問題なく収束します。他方、分子動力学シミュレーションにおいて保存量を保存させるには、さらに厳しい収束判定を採用する必要があります。

図 7-3 に、 SiO_2 に対して収束判定を変化させながら力の計算を行った結果を例としてしめます。この図から、力を収束させるためには 10^{-10} hartree 以上の、比較的厳しい収束判定が要求されることが分かります。

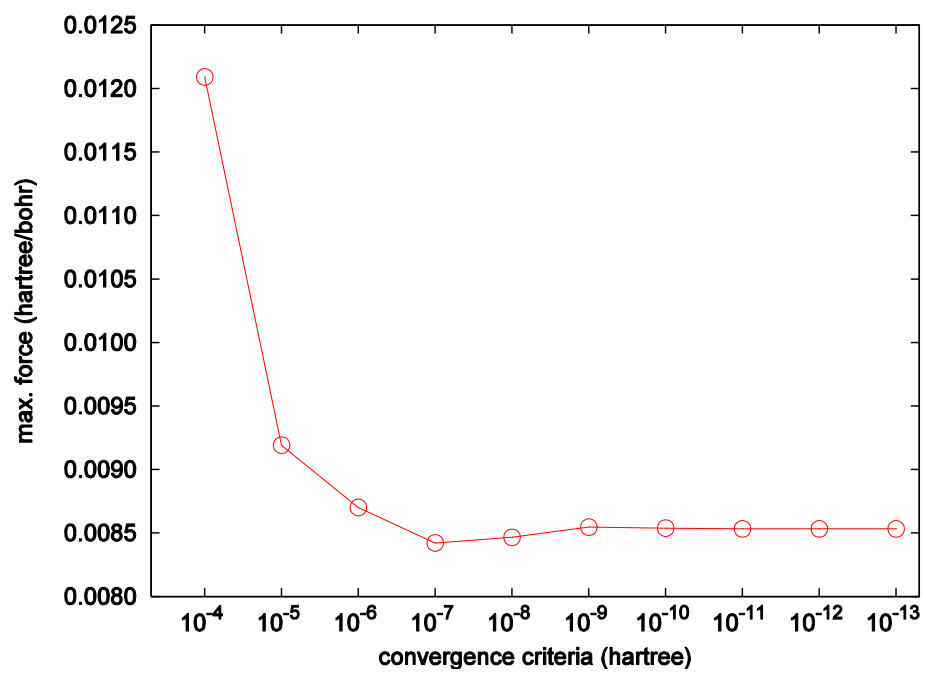


図 7-3 SiO₂ の、収束判定と力の最大値の関係

7.1.4 ベンチマーク計算例（波動関数ソルバーの比較）

7.1.4.1 FCC-Cu

FCC-Cu について、各波動関数ソルバーがどの程度のパフォーマンスを示すか確かめてみましょう。なお、ここで紹介する計算の入力データは、以下のディレクトリのサブディレクトリにあります。

`samples/sol_cmix_test/Cu`

（1）入力データ

まず、波動関数ソルバー以外の入力データです。

```
Control{
  condition = initial
  cpumax = 1 day
}
accuracy{
  cutoff_wf = 25.0 rydberg
  cutoff_cd = 225.0 rydberg
  num_bands = 10
  ksampling{
    mesh{
      nx = 10
      ny = 10
      nz = 10
    }
  }
  scf_convergence{
    delta_total_energy = 1.e-10 hartree
    succession = 3
  }
  initial_wavefunctions = atomic_orbitals
  initial_charge_density = atomic_charge_density
}
structure{
  unit_cell_type = primitive
  unit_cell{
    !#units bohr
    a_vector = 0.0000000 3.4704637 3.4704637
    b_vector = 3.4704637 0.0000000 3.4704637
    c_vector = 3.4704637 3.4704637 0.0000000
  }
  symmetry{
    method = automatic
    tspace{
      lattice_system = fcc
    }
    sw_inversion = on
  }
  atom_list{
    atoms{
      !#tag rx ry rz weight element mobile
      0.000 0.000 0.000 1 Cu 0
    }
  }
  element_list{
    #tag element atomicnumber
    Cu 29
  }
}
wavefunction_solver{
  次節参照
}
charge_mixing{
  mixing_methods{
    !#tag method rmxs rmxe itr var prec istr nbmix update
    broyden2 0.60 0.60 * * on 3 15 RENEW
  }
}
printlevel{
```

```

    base = 1
}

```

以下に波動関数ソルバーの設定例を示します。

- 行列対角化

```

wavefunction_solver{
  solvers{
    !#tag    id sol    till_n dts dte itr var prec cmix submat
          1 matrixdiagon -1 * * * * on 1 off
  }
}

```

- lm+msd, 部分空間対角化は波動関数更新後

```

wavefunction_solver{
  solvers{
    !#tag    id sol    till_n dts dte itr var prec cmix submat
          1 lm+msd 1 * * * * on 1 on
  }
  submat{
    before_renewal=off
  }
}

```

- lm+msd, 部分空間対角化は波動関数更新前

```

wavefunction_solver{
  solvers{
    !#tag    id sol    till_n dts dte itr var prec cmix submat
          1 lm+msd 1 * * * * on 1 on
  }
  submat{
    before_renewal=on
  }
}

```

- lm+msd → rmm3, 部分空間対角化は波動関数更新後

```

wavefunction_solver{
  solvers{
    !#tag    id sol    till_n dts dte itr var prec cmix submat
          1 lm+msd 1 * * * * on 1 on
          2 rmm3 -1 * * * * on 1 on
  }
  rmm{
    edelta_change_to_rmm = 5.0e-3
  }
  submat{
    before_renewal=off
  }
}

```

- lm+msd → rmm3, 部分空間対角化は波動関数更新前

```

wavefunction_solver{
  solvers{
    !#tag    id sol    till_n dts dte itr var prec cmix submat
          1 lm+msd 1 * * * * on 1 on
          2 rmm3 -1 * * * * on 1 on
  }
  rmm{
    edelta_change_to_rmm = 5.0e-3
  }
  submat{
    before_renewal=on
  }
}

```

- Davidson → rmm3, 部分空間対角化は波動関数更新後

```

wavefunction_solver{
  solvers{
    !#tag    id sol    till_n  dts  dte  itr  var  prec cmix submat
          1  davidson   1      *    *    *    *    off 1 off
          2   rmm3    -1      *    *    *    *    on  1  on
  }
  rmm{
    edelta_change_to_rmm = 5.0e-3
  }
  submat{
    before_renewal=off
  }
}

```

- Davidson → rmm3, 部分空間対角化は波動関数更新前

```

wavefunction_solver{
  solvers{
    !#tag    id sol    till_n  dts  dte  itr  var  prec cmix submat
          1  davidson   1      *    *    *    *    off 1 off
          2   rmm3    -1      *    *    *    *    on  1  on
  }
  rmm{
    edelta_change_to_rmm = 5.0e-3
  }
  submat{
    before_renewal=on
  }
}

```

- Davidson

```

wavefunction_solver{
  solvers{
    !#tag    id sol    till_n  dts  dte  itr  var  prec cmix submat
          1  davidson  -1      *    *    *    *    off 1 off
  }
}

```

(2) 結果

ベンチマークテストの計算結果を図 2 に、計算時間を表 2 に示します。Intel Fortran Compiler 11.1 for Linux でコンパイルして、2.4GHz の Opteron280 プロセッサを搭載したクラスターマシンにおいて k 点 4 並列で計算を行いました。

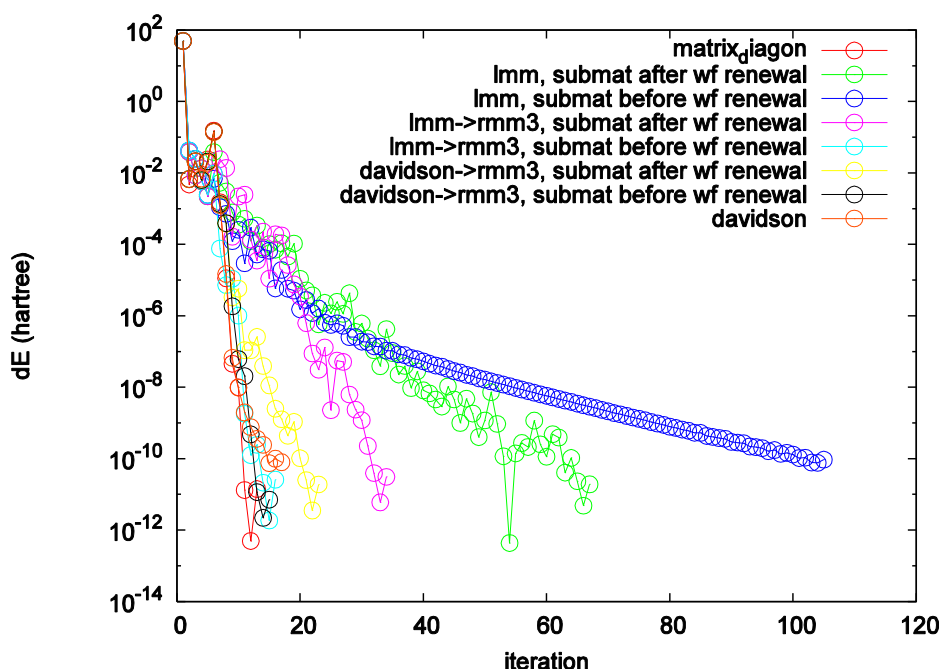


図 7-4 波動関数ソルバーによる収束の速さの比較

表 7.1 は波動関数ソルバーによる計算時間の比較です。繰り返し回数は、電荷の混合の回数です。また、計算時間は Opteron 280 2.4GHz のクラスターマシンで、k 点 4 並列で行った場合の結果です。あくまで参照値とお考えください。

表 7.1 波動関数ソルバーによる計算時間の比較。

手法	繰り返し回数	計算時間
行列対角化法	13 回	19.2 秒
lm+msd, 部分空間対角化は波動関数更新後	67 回	22.2 秒
lm+msd, 部分空間対角化は波動関数更新前	105 回	32.4 秒
lm+msd → rmm3, 部分空間対角化は波動関数更新後	34 回	12.4 秒
lm+msd → rmm3, 部分空間対角化は波動関数更新前	16 回	8.4 秒
Davidson → rmm3, 部分空間対角化は波動関数更新後	23 回	11.2 秒
Davidson → rmm3, 部分空間対角化は波動関数更新前	15 回	9.5 秒
Davidson	17 回	11.8 秒

横軸は繰り返し回数で、縦軸は収束値からの相対的なエネルギーです。SCF 計算では、変分原理が成り立つので、エネルギーが低いほど、正確な値となります。行列対角化法は、繰り返し回数で見た収束は速いのですが、1 回あたりの計算量は一般に大きく、とくに、系のサイズが大きくなると、この手法は実際上適用できなくなります。全体的には、途中で rmm3 法へ移行するケースが速く収束しています。また、rmm3 法は部分空間対角化を波動関数更新前に適用する方がより速く収束しています。

計算する系に依存して、収束の速さ、安定性は変化します。そこで、その都度、最適な計算手法を選択する事をお勧めいたします。比較的安定で収束の速い手法として、LM+MSD→RMM3 法、Davidson 法、Davidson→RMM3 法が多くの場合推奨されます。また、RMM3 法を利用する場合部分空間対角化は波動関数更新前に適用した方がよい場合が多いです。Davidson 法を利用する場合、前処理(precon)は off としておいた方がよいでしょう。

7.1.4.2 Fe(100) 表面

次に、スピンを考慮した計算の例として Fe(100)表面の計算例を紹介します。この問題では、波動関数ソルバーは固定とし、電荷密度混合法を変更したテストを行います。ここで紹介する計算の入力データは、以下のディレ

クトリのサブディレクトリーにあります。

`samples/sol_cmix_test/Fe100`

(1) 入力データ

まず、電荷密度混合法以外の入力データです。

```
control{
  condition = initial
  max_iteration = 200
}
accuracy{
  num_bands = 52
  ksampling{
    method=monk
    mesh{
      nx = 6
      ny = 6
      nz = 1
    }
  }
  cutoff_wf = 30 rydberg
  cutoff_cd = 300 rydberg
  initial_wavefunctions = atomic_orbitals
  initial_charge_density = atomic_charge_density
  scf_convergence{
    delta_total_energy = 1e-9
    succession = 3
  }
  force_convergence{
    max_force = 0.0005 hartree/bohr
  }
}
structure{
  atom_list{
    atoms{
      #tag    element    rx    ry    rz    mobile    weight
      Fe     0.5      0.5    0    off     1
      Fe     0        0    0.0948333333333333 off     2
      Fe     0        0    0.2845    off     2
      Fe     0.5      0.5    0.1896666666667 off     2
    }
  }
  ferromagnetic_state{
    sw_fix_total_spin=off
    total_spin=14
    spin_fix_period=5
  }
  unit_cell{
    a_vector = 5.3762704477 0.0 0.0
    b_vector = 0.0 5.3762704477 0.0
    c_vector = 0.0 0.0 28.3458898822
  }
  element_list{
    #tag    element    atomicnumber    mass    zeta    deviation
    Fe     26      101802.230406    0.375    1.83
  }
  symmetry{
    method = automatic
    sw_inversion = on
  }
  magnetic_state=ferro
}
structure_evolution{
  method = gdiis
  gdiis{
    initial_method = cg
    c_forc2gdiis = 0.005 hartree/bohr
  }
}
```

```

wavefunction_solver{
  solvers{
    #tag      sol      till_n      prec      cmix      submat
      davidson  1      off      1      off
      rmm3      -1      on      1      on
  }
  rmm{
    edelta_change_to_rmm = 5e-3 hartree
  }
  submat{
    before_renewal = on
  }
}
charge_mixing{
  次節参照
}
printoutlevel{
  base = 1
}

```

以下に電荷密度混合法の設定例を示します。

電荷密度の和と差で異なる混合比を採用，混合アルゴリズムは Broyden2 法(case0)

```

charge_mixing{
  spin_density_mixfactor=4
  mixing_methods{
    #tag      no      method      rmxs      rmxe      itr      var      prec      istr      nbmix      update
      1      broyden2      0.1      0.1      40      linear      on      3      5      renew
  }
}

```

電荷密度の和と差で異なる混合比を採用，混合アルゴリズムは Pulay 法(case1)

```

charge_mixing{
  spin_density_mixfactor=4
  mixing_methods{
    #tag      no      method      rmxs      rmxe      itr      var      prec      istr      nbmix      update
      1      pulay      0.1      0.1      40      linear      on      3      15      renew
  }
}

```

電荷密度の和と差で同じ混合比を採用，混合アルゴリズムは Broyden2 法(case2)

```

charge_mixing{
  spin_density_mixfactor=1
  mixing_methods{
    #tag      no      method      rmxs      rmxe      itr      var      prec      istr      nbmix      update
      1      broyden2      0.1      0.1      40      linear      on      3      15      renew
  }
}

```

電荷密度の和と差で同じ混合比を採用，混合アルゴリズムは Pulay 法(case3)

```

charge_mixing{
  spin_density_mixfactor=1
  mixing_methods{
    #tag      no      method      rmxs      rmxe      itr      var      prec      istr      nbmix      update
      1      pulay      0.1      0.1      40      linear      on      3      15      renew
  }
}

```

(2) 結果

本ベンチマークの結果を，表 3 にまとめました。同じ電子状態へ収束していることを確認するため，得られた全エネルギーもあわせて表示しています。この例では，「電荷密度の和と差で同じ混合比を採用，混合法は Pulay 法」(case3)で最少の回数で収束解を得ることができました。多くの場合 case3 の設定によって少ない計算回数で収束解が得られますが，問題によっては Broyden2 法を採用した方が良い場合もありますし，電荷密度の差の

混合比を大きくした方がよい場合もあります。さらに、スピンを考慮した計算の場合、初期スピン分極やスピンを固定するか否かなどによっても収束性は変化します。収束しづらい（しない）場合、上記の設定例などを参考に、最適な電荷密度混合法の設定を行ってください。

表 7.2 Fe (100) 表面の収束に至るまでの SCF 計算回数と到達した全エネルギー

	SCF 回数	全エネルギー (ha.)
case0	36	-153.877775988322
case1	32	-153.877775991437
case2	34	-153.877775825592
case3	29	-153.877775990755

7.1.5 波動関数ソルバーおよび電荷密度混合法の自動設定

PHASE に搭載されている波動関数ソルバーには、MSD 法、lm+MSD 法、Davidson 法、CG 法、RMM 法、直接対角化法などの基本ソルバーと補助ソルバーとしての subspace rotation があります。さらに、電荷密度ミキサーとして単純混合法、Pulay 法、Broyden による 2 番目の方法などを搭載しています。これらを、問題に応じて適切に組み合わせることによって高速な収束が期待できます。しかし、このように問題に応じて適切に組み合わせるのは非常に手間がかかる作業です。そこで、PHASE には、適切な波動関数ソルバーや電荷密度混合法をプログラムが自動的に選択する機能が備わっています。ここでは、この機能の利用方法を説明します。この機能によって様々な系を収束させることができていますが、もしなかなか収束させられない場合は、手動で波動関数ソルバーや電荷密度ミキサーの設定を行うようにしてください。

この機能を利用することによって、様々な系で安定に収束解が得られていることを確認しています。例として、鉄(100)表面の収束性を示したものを次の図に示します。

入力パラメーターファイルに波動関数ソルバーおよび電荷密度混合法の指定がない場合の収束性を PHASE v1100 と PHASE/0 2014 で比較しました。計算は磁性を考慮して行いました。結果は下図の通りです。

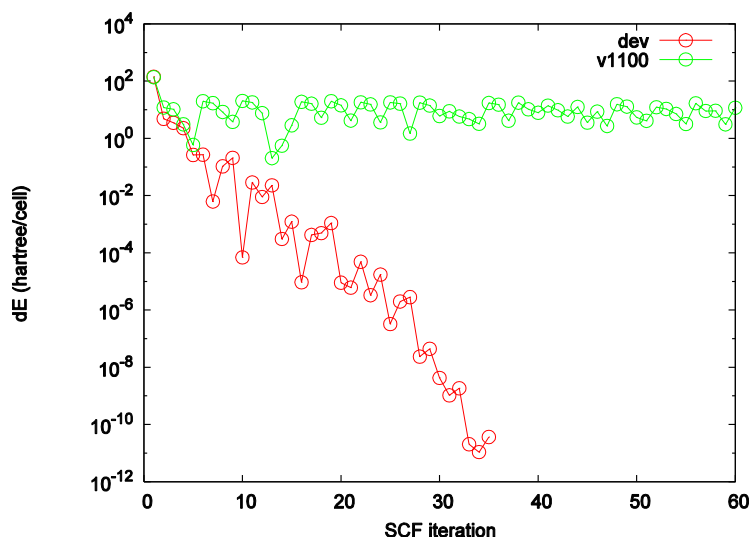


図 7-5 波動関数ソルバーおよび電荷密度ミキサーの自動設定

赤線が本機能の結果、緑線が PHASE v1100 を利用した結果です。

図から明らかなように、PHASE v1100 を利用すると波動関数ソルバー・電荷密度混合法の設定をしない場合は本例題を収束させることができませんが、本機能を利用すると収束させることができています

7.2 構造最適化

7.2.1 構造最適化手法

7.2.1.1 計算例

構造最適化の各アルゴリズムがどのように振る舞うかを調べるため、以下に挙げる系の構造最適化を搭載されているアルゴリズムを利用して実施しました。

- ・ ケース 1: cis 型のジクロロシクロヘキサン
- ・ ケース 2: ルチル型 TiO₂
- ・ ケース 3: SiO₂
- ・ ケース 4: Si(001) 表面

これらの入力は、以下のディレクトリ下のサブディレクトリにあります。

`samples/strevl_test`

力の収束判定は、すべてのケースで 10^{-4} hartree/bohr としました（この条件は、比較的厳しい収束判定条件です）。また、電子状態計算の収束判定はすべてのケースで 10^{-10} hartree 1 回としました。原子配置の更新は、最大 200 回行い、それでも収束しないケースは未収束と見なしました。

各々の構造最適化のアルゴリズムは、以下のように設定しました。

(1) quenched MD 法

```
structure_evolution{
  method = quench
}
```

(2) cg 法

```
structure_evolution{
  method = cg
}
```

(3) gdiis 法

```
structure_evolution{
  method = gdiis
  gdiis{
    initial_method = cg
    c_forc2gdiis = 0.01 hartree/bohr
  }
}
```

このように設定すると、まずは cg 法のアルゴリズムに従って最適化が進行します。原子に働く力の最大値が `c_forc2gdiis` で指定する値よりも小さくなった時点で gdiis 法に切り替わります。ただし、最初の 3 回は原子に働く力の最大値に関わらずに cg 法が採用されます。

(4) bfgs 法

```
structure_evolution{
  method = bfgs
  gdiis{
    initial_method = cg
    c_forc2gdiis = 0.01 hartree/bohr
  }
}
```

このように設定すると、まずは **cg** 法のアルゴリズムに従って最適化が進行します。原子に働く力の最大値が **c_forc2gdiis** で指定する値よりも小さくなった時点で **bfgs** 法に切り替わります。ただし、最初の 3 回は原子に働く力の最大値に関わらずに **cg** 法が採用されます。**bfgs** 法の詳細設定は **gdiis** ブロックで行い、また各変数の意味も **gdiis** 法の場合と全く同じです。

7.2.1.2 計算結果

ベンチマークの結果を表 7.3 にまとめました。この結果、**quenched MD** 法は収束が遅いことが分かります。今回のケースでは時間刻みとしてデフォルト値 (100 au) を採用しましたが、この値を調整することによって収束が改善される可能性はあります。**GDIIS** 法は **SiO₂** のケースでは最も速く収束していますが、それ以外のケースではあまり有効に作用していません。**gdiis** 法の振る舞いは、**c_forc2gdiis** パラメーターをより小さなものにしたリ、電子状態計算の収束判定をより厳しくすることによって改善される可能性はあります。**cg** 法は比較的安定に最適化ができています。**bfgs** 法は、すべてのケースで収束し、また平均的に少ない回数で最適化を行うことができました。

表 7.3 構造緩和法の違いによる収束にいたる回数の比較。200 回の更新を経ても力の最大値が 10^{-4} 以下とならなかったケースは未収束とした。ケース 1 がジクロロシクロヘキサン、ケース 2 がルチル型 **TiO₂**、ケース 3 が **SiO₂**、ケース 4 が **Si(001)** 表面に相当する。

	ケース 1	ケース 2	ケース 3	ケース 4
quenched MD	未収束	115	166	未収束
cg	195	62	28	124
GDIIS	未収束	71	13	176
BFGS	87	38	16	67

7.2.2 構造更新時の電荷密度、波動関数の予測更新（収束性の向上）

PHASE には、構造最適化や分子動力学シミュレーションを行っている際に、波動関数や電荷密度を原子配置の変化に合わせて“補外”することによって収束性を向上させる機能が備わっています。補外は、文献[2]で紹介されている方法によって行っています。いくつかの系において本機能を適用した結果を示します。いずれのケースにおいても、おおむね高速化されます。

鉄粒界

	no prediction	charge0	charge1
原子配置更新回数	11	11	11
SCF 回数	245	185	163

ジクロロシクロヘキサン分子

	no prediction	charge0	charge1
原子配置更新回数	25	25	25
SCF 回数	481	330	310

SiO₂ 結晶

	no prediction	charge0	charge1
原子配置更新回数	16	16	16
SCF 回数	193	144	136

Si (100) 表面

	no prediction	charge0	charge1	charge1+wf
--	---------------	---------	---------	------------

iteration_ionic	21	21	21	21
SCF 回数	324	200	187	157

上記表中、charge0 は sw_charge_predictor = on, charge1 は sw_charge_predictor = on と sw_extrapolate_charge = on, charge1+wf は sw_charge_predictor = on , sw_extrapolate_charge = on, sw_wf_predictor = on に相当

Si 64 原子 300K MD 50 回

	no prediction	charge0	charge1	charge1+wf
SCF 回数	613	413	387	231

上記表中、charge0 は sw_charge_predictor = on, charge1 は sw_charge_predictor = on と sw_extrapolate_charge = on, charge1+wf は sw_charge_predictor = on , sw_extrapolate_charge = on, sw_wf_predictor = on に相当

- [2] T. Arias, M. C. Payne and J. D. Joannopoulos, “*Ab initio* molecular-dynamics techniques extended to large-length-scale systems”, Physical Review B **45**, 1538 (1992).

7.3 PHASE/0 の単位系

PHASE/0 において利用される単位は、原則としてハートリー原子単位系です。ここでは、ハートリー原子単位系からそのほかの単位に変換する際の 変換係数を記述します。結果の解析の際にご活用ください。

エネルギー	$1 \text{ hartree} = 2 \text{ rydberg} = 27.21139615 \text{ eV} = 4.359745836 \times 10^{-18} \text{ J}$
長さ	$1 \text{ bohr} = 0.5291772480 \text{ \AA} = 0.5291772480 \times 10^{-10} \text{ m}$
質量	$1 \text{ au mass} = \text{電子の質量} = 9.1094 \times 10^{-31} \text{ kg}$
体積	$1 \text{ au volume} = 0.1481847426 \text{ \AA}^3 = 1.48184726 \times 10^{-29} \text{ m}^3$
速度	$1 \text{ au velocity} = 2.187691417 \times 10^{-2} \text{ \AA/s} = 2.187691417 \times 10^8 \text{ m/s}$
力	$1 \text{ hartree/bohr} = 51.42208259 \text{ eV/\AA} = 8.238725025 \times 10^{-8} \text{ N}$
時間	$1 \text{ au time} = 2.418884327 \times 10^{-2} \text{ fs} = 2.418884327 \times 10^{-17} \text{ s}$
ストレス	$1 \text{ au stress} = 2.903628623 \times 10^9 \text{ atm} = 2.942101703 \times 10^{13} \text{ Pa}$
密度	$1 \text{ au density} = 1.23013834 \times 10^4 \text{ amu/\AA}^3 = 9.1093897 \times 10^{-4} \text{ g/cm}^3 = 9.1093897 \times 10^{-1} \text{ kg/m}^3$

8. PHASE/0 のインストール

8.1 動作環境

PHASE/0 は、PC から最先端のスーパーコンピュータの様々な計算機環境で動作します。

PHASE/0 プログラムは `fortran` (`fortran90` および `fortran77`) と `C` で記述されています。これらのコンパイラが使える計算機システムが必要です。並列計算をする場合には `MPI` ライブラリがインストールされている必要があります。

必要 (利用可能) なソフトウェア、ライブラリ

- Fortran90 コンパイラ、C コンパイラ (必須)
- MPI ライブラリ (並列計算に必須)
- 行列演算ライブラリ LAPACK, BLAS (オプション)
- FFT ライブラリ FFTW (オプション)
- Perl (オプション)・・・PHASE ツールで必要
- Gnuplot (オプション)・・・PHASE ツールで必要
- Python (オプション)・・・PHASE ツール, ASE で必要

PHASE の動作確認を行っている計算機環境を以下に示します。

PHASE/0 が動作する計算機環境

計算機環境	コンパイラー	利用可能ライブラリ
Linux	GNU Compiler Intel Compiler	LAPACK, BLAS, ScaLAPACK MKL FFTW3
Intel 版 Mac OS X	GNU Compiler Intel Compiler	MKL FFTW3
NEC SX Series	Fortran90/SX	Mathkeisan(LAPACK) ASL(FFT)
Fujitsu FX10, FX100	Fujitsu Compiler	

- MPI ライブラリは、OpenMPI, IntelMPI, SGI MPT などに対応しています。
- GNU Compiler (`gfortran`, `gcc`) は、バージョン 4.4 以降を使用してください。
- MKL の FFTW3 ラッパーを介して MKL の FFT を利用することもできます。
- ASL の FFTW3 ラッパーを介して ASL の FFT を利用することもできます。

8.2 2次元版および3次元版について

PHASE/0 は、バンドおよび **k** 点の 2 軸で並列化を施した 2 次元版と、バンド、**k** 点に加え **G** 点の並列軸を加えた 3 次元版があります。低並列時では 2 次元版の方が高速に動作することが多いですが、高並列時は 3 次元版に軍配があがります。ただし、一部機能は 2 次元版でしか利用することはできません。

8.3 インストール方法

Linux 環境を例にしてインストール方法を説明します。ここでは、Linux 環境に Intel Fortran compiler がインストールされていることを想定します。別のコンパイラを利用する場合はコンパイラの選択のときに、そのコンパイラを選択してください。また、MPI ライブラリとして、OpenMPI を使用することを仮定しています。プログラムモデルの選択のときに「Serial」を選択すれば MPI ライブラリを使用しないでプログラムを作成することもできます。

まず、PHASE/0 パッケージ phase0_2019.01.tar.gz を PHASE/0 をインストールするディレクトリに展開してください。

```
$ tar xzf phase0_2019.01.tar.gz
```

ディレクトリ phase0_2019.01 に移り、インストーラーを実行してください(2 次元版は install.sh, 3 次元版は install_3d.sh ; 以下は 2 次元版の例)

```
$ cd phase0_2019.01
$ ./install.sh
```

```
=== PHASE installer ===
```

```
Do you want to install PHASE? (yes/no) [yes]
```

インストールするかどうか聞いてきますので、何も入力せずに Enter キーを押してください。

```
Supported platforms
```

- 0) GNU Linux (IA32)
- 1) GNU Linux (EM64T/AMD64)
- 2) NEC SX Series
- x) Exit

```
Enter number of your platform. [0]
```

対応する環境の一覧が表示されますので、「GNU Linux (IA32)」に対応する 0 を入力して、Enter キーを押してください。

```
Supported compilers
```

- 0) GNU compiler collection (gfortran)
- 1) Intel Fortran compiler
- x) Exit

```
Enter number of a desired compiler. [0]
```

対応するコンパイラーの一覧が表示されますので、「Intel Fortran compiler 9.x」に対応する 2 を入力して、Enter キーを押してください。

```
Supported programming-models
```

- 0) Serial
- 1) MPI parallel
- x) Exit

```
Enter number of a desired programming-model. [0]
```

対応するプログラムモデルの一覧が表示されますので、「MPI parallel」に対応する 1 を入力して、Enter キーを押してください。

```
Supported MPI libraries
```

- 0) Open MPI
- 1) SGI MPT
- 2) Intel(R) MPI
- x) Exit

```
Enter number of a desired MPI library. [0]
```

対応する MPI ライブラリの一覧が表示されますので、「Open MPI」に対応する 0 を入力して、Enter キーを押してください。

```
Supported BLAS/LAPACK
```

- 0) Netlib BLAS/LAPACK
- 1) Intel Math Kernel Library (MKL)

x) Exit Enter number of a desired library. [0]
<p>対応する BLAS/LAPACK ライブラリの一覧が表示されますので、「Netlib BLAS/LAPACK」に対応する 0 を入力して、Enter キーを押してください。</p>
Supported FFT libraries 0) Built-in FFT subroutines 1) FFTW3 library x) Exit Enter number of a desired library. [0]
<p>対応する FFT ライブラリの一覧が表示されますので、「Built-in FFT subroutines」に対応する 0 を入力して、Enter キーを押してください。</p>
Do you want to edit the makefile that has been generated? (yes/no/exit) [no]
<p>作成された Makefile を編集するかどうか聞いてきます。Makefile の確認や編集を行う必要がなければ、何も入力せずに Enter キーを押してください。</p>
Do you want to make PHASE now? (yes/no) [yes]
<p>PHASE のコンパイルとインストールを開始するかどうか聞いてきます。何も入力せずに Enter キーを押して、PHASE のコンパイルとインストールを始めてください。</p>
PHASE was successfully installed. Do you want to check the installed programs? (yes/no) [no]
<p>PHASE が正常にインストールされたことを告げるメッセージの後、プログラムのテスト計算を実行するかどうか聞いてくるので、必要があれば yes を入力し、Enter キーを押してください。テスト計算をしないならば、no を入力して Enter キーを押してください。テスト計算を実行して以下のような出力が得られれば問題ありません。</p>
Do you want to check the installed programs? (yes/no) [no] yes Checking total-energy calculation. Total energy : -7.897015156331 Hartree/cell Reference : -7.897015156332 Hartree/cell Checking band-energy calculation. Valence band maximum : 0.233846 Hartree Reference : 0.233846 Hartree

MPI プログラムの実行に用いる mpirun や mpiexec などのコマンドを用いて実行します。

\$HOME/phase0_2019.01/bin を環境変数 PATH に追加しておく、PHASE/0 のプログラムのパスを指定せずに実行でき便利です。

Bourne shell(ボーンシェル)系であれば、\$HOME/.bashrc などに PATH を記述します。

```
export PATH=$HOME/phase0_2019.01/bin:$PATH
```

C shell(シーシェル)系であれば、\$HOME/.cshrc に PATH を記述します。

```
setenv PATH $HOME/phase0_2019.01/bin:$PATH
```

MPI ライブラリの bin ディレクトリにも必ずパスを通すようにしてください。

Bourne shell(ボーンシェル)系であれば、\$HOME/.bashrc などに PATH を記述します。

```
export PATH=$HOME/openmpi/bin:$PATH
```

C shell(シーシェル)系であれば、\$HOME/.cshrc に PATH を記述します。

```
setenv PATH $HOME/openmpi/bin:$PATH
```

以下のようにして、PHASE を実行します。

```
$ mpirun -np 2 phase ne=1 nk=2
```

9. PHASE/0 プログラム、ツールの実行方法

9.1 プログラム phase

9.1.1 プログラム phase の実行

PHASE は SCF 計算、分子動力学法計算を行います。また収束した電荷密度分布から状態密度やバンド分散を計算することができます。

入力パラメータファイル、擬ポテンシャルファイルを実行ディレクトリに置きます。file_names.data を使用する場合には、それも同じディレクトリに置いてください。

1 プロセッサ（1 コア）の逐次計算を行う場合には、次のようにプログラム phase を実行します。”../../phase0_2019.01/bin/”は、PHASE がインストールされているディレクトリです。

```
% ../../phase0_2019.01/bin/phase
```

並列計算を行う場合には、お使いの計算機の利用する MPI ライブラリの実行コマンドを使用します。詳細はお使いの計算機システムのマニュアルを参照ください。一般的なコマンドは mpirun です。

```
% mpirun -np NP ../../phase0_2019.01/bin/phase ne=NE nk=NK (2次元版)
% mpirun -np NP ../../phase0_2019.01/bin/phase.3d ne=NE nk=NK ng=NG (3次元版)
```

ここで、NP は MPI プロセス数、NE はバンド並列数、NK は k 点並列数です。3 次元版の場合の NG は G 点並列数です。

9.1.2 プログラム phase の並列計算オプション

9.1.2.1 バンド並列、k 点並列

並列計算（バンド並列、k 点並列）では、バンド並列数 NE、k 点並列数 NK を指定します。NP = NE × NK という関係が成立している必要があります。ne, nk という引数は省略することも可能です。その場合は NE = NP, NK = 1 が採用されます。

```
% mpirun -np NP ../../phase0_2019.01/bin/phase ne=NE nk=NK
```

通常、バンド並列よりも k 点並列の方が効率が良いです。したがって、可能な場合は k 点並列数を大きくすると良いと考えられます。ただし、Brillouin 領域内にサンプルする k 点数は系が大きくなるほど少なくとも充分になり、その k 点数が必ずしも利用できるプロセス数で割り切れるわけではない (MPI プロセス数がサンプリング k 点よりも大きくなる) という点に注意が必要です。k 点数よりも NK の値が大きいとエラーになります。また、k 点数が NK で割り切れない場合は理想的な並列効率が得られません。そこで、必要に応じてバンド並列も組み合わせて計算を実行してください。

9.1.2.2 レプリカ並列

NEB 法、拘束条件付きダイナミクス、メタダイナミクスなどの機能によっては“レプリカ並列”が利用できる場合があります。レプリカ並列を実行するには以下のコマンドを利用します。

```
% mpirun -np NP ../../phase0_2019.01/bin/phase nr=NR ne=NE nk=NK
```

NR はレプリカ並列数です。NP = NR × NE × NK という関係が成立している必要があります。レプリカ並列の効率は k 点並列よりも更に良いですが、k 点並列と同じ注意が必要です。また、“もっとも収束の遅いレプリカ”が律速となるので、実効的には必ずしも効率的とは限りません。

9.1.3 プログラム phase 3 次元並列 (G 点並列) 版

PHASE/0 はバンドと k 点の 2 軸並列に対応していますが、平面波の G 成分の並列化にも対応しています。

3 軸並列版も、2 軸並列版と同様インストーラーによってコンパイルすることができます。

```
% ./install_3d.sh
```

実行は、次の例のように行います。

```
% mpirun -np NP ../../phase0_2019.01/bin/phase.3d ng=NG ne=NE nk=NK
```

ここで、NP は総 MPI プロセス数、NG は G 点並列数、NE はバンド並列数、NK は k 点並列数を意味します。NG と NE と NK の積は、総 MPI プロセスの数に等しい必要があります。

9.1.4 3 次元版のオプションについて

3 次元版は、以下のような設定を施すことによって、おもに低並列時に高速になる場合があります。

(1) FFT を非並列で処理する

3 次元版は FFT を並列で処理します。これをあえて非並列で処理することによって、G 点並列数が少ない場合に高速になる場合があります。このオプションは以下の要領で有効にすることができます。

```
control{  
    sw_serial_fft = on  
}
```

特に ng=1 とする場合は高速化が期待できるオプションです。

(2) 電荷密度の処理用にコミュニケーターを追加する

3 次元版は、電荷密度を波動関数と同じコミュニケーターで扱います。以下の設定を施すことによって電荷密度に専用のコミュニケーターを割り当てることができます。

```
control{  
    sw_communicator_for_chg = on  
}
```

G 並列数が少ない場合に、おもに交換相関相互作用の処理が高速化されます。

9.2 プログラム ekcal

状態密度計算、バンド計算において、 k 点の個数が多い場合に使うプログラムとして **ekcal** があります。SCF 計算の計算結果の電荷密度を入力として計算できます。

SCF 計算の計算結果の電荷密度ファイル **nfchgt.data** を実行ディレクトリにコピーします。または、入出力ファイル設定ファイル **file_names.data** において、**F_CHG** に SCF 計算で得られた電荷密度ファイル指定します。

バンド構造計算においては、サンプリング k 点の設定ファイル **kpoint.data** を用意します。

次のようにプログラム **ekcal** を実行します。”**phase0_2019.01/bin/**”は、PHASE がインストールされているディレクトリです。

```
% ../../phase0_2019.01/bin/ekcal
```

ekcal プログラムは 2 次元版にしか用意されていませんが、入力パラメーターファイルに以下のような設定を加えることによって **phase** を **ekcal** と同じように動作させることができます。

```
control{
  fixed_charge_option{
    kparallel = one_by_one
  }
}
```

9.3 プログラム epsmain

電子系の誘電関数の計算に利用するプログラムが **epsmain** です。その動作は **ekcal** とほぼ同じですが、入力パラメーターの設定に応じて誘電関数計算用の処理が行われる点が異なります。3 次元版のバイナリー名は **epsmain.3d** です。

9.4 プログラム tdlrmain

線形応答時間依存密度汎関数法による励起スペクトル計算に利用するプログラムが **tdlrmain** です。3 次元版は用意されていません。

9.5 状態密度図作成ツール dos.pl

状態密度図の作成

PHASE あるいは EKCAL によって状態密度データを出力させることが出来ます。 それについては、たとえば 3.10.1 をご覧ください。 その状態密度データ dos.data を可視化するプログラムが dos.pl example の dos.data を work にコピーします。

```
$ cp ../example/dos.data
```

間違いなく dos.data がコピーされていることを ls で確認してください。

を用いて、この状態密度データ dos.data を可視化してみましょう。次のようにコマンドを入力して下さい。

```
$ dos.pl dos.data -erange
```

こうすると、EPS ファイル density_of_states.eps が生成されます。 UNIX 環境で、これを見るには ghostview や gv などを必要とします。 \$ ghostview

または \$ gv

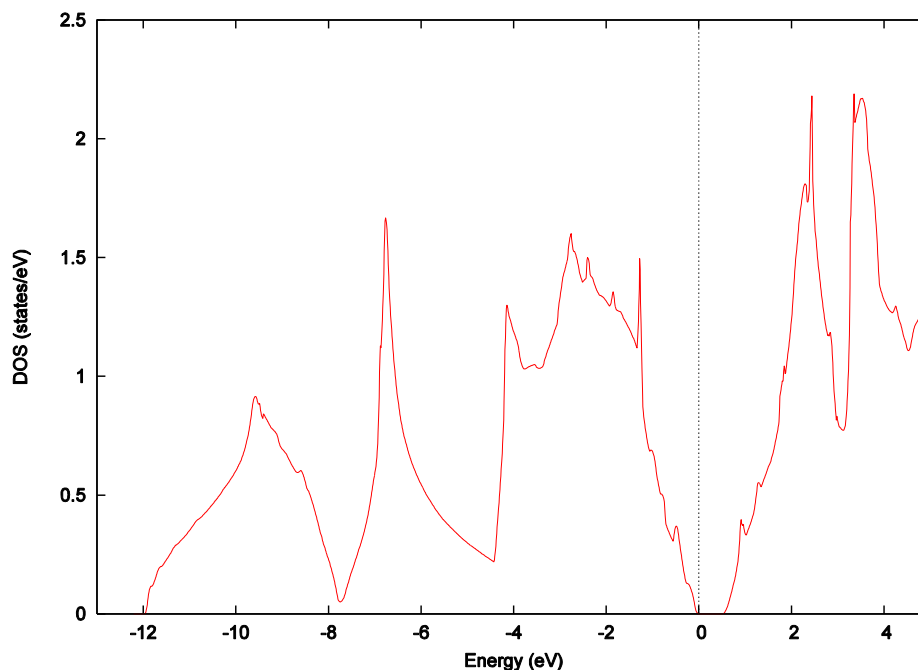


図 9-1 バルク Si の状態密度図

dos.pl を実行するときに状態密度データ dos.data の後に付加した -erange は表示するエネルギーの範囲を 制御するオプション、-color はカラー出力を行うためのオプションです。

9.5.1 dos.pl のオプション

なにも付加せずに dos.pl を実行すると利用方法が表示されます。

```
$ dos.pl
```

```
Version: 3.00
Usage: dos.pl DosData -erange=Emin,Emax -einc=dE -dosrange=DOSmin,DOSmax -dosinc=dDOS
-ttitle=STRING -with_fermi -width=SIZE -font=SIZE -color -mode={total|layer|atom|projected}
-epsf={yes|no} -data={yes|no}
```

DosData に状態密度データが記録されたファイル(通常 dos.data)を指定します。その後には作図を制御するオプションを指定します。

-erange=Emin, Emax	表示するエネルギーの範囲を eV 単位で指定する。 たとえば、-10 eV から 5 eV まで表示したい場合は、 -erange=-10, 5 とします。指定をしないと、データの最小値・最大値から自動的に決定されます。
-einc=dE	目盛りの間隔を指定する。 たとえば、2eV 間隔に目盛りをふりたいなら、 -einc=2 とします。
-dosrange=DOSmin, DOSmax	表示する状態密度の範囲を変える。 たとえば、0 states/eV から 12 states/eV まで表示したい場合は、 -dosrange=0, 12 とします。
-dosinc=dDOS	縦軸(状態密度)の目盛りの間隔を指定する。 たとえば、2 states/eV 間隔に目盛りをふりたいなら、 -dosinc=2 とします。
-ttitle=STRING	グラフにタイトルを付けたいときに設定する。 たとえば、タイトルを「Total DOS」とするなら、 -ttitle="Total DOS" とします。
-with_fermi	デフォルトでは描かないフェルミレベルまたは価電子帯上端のエネルギーレベルを描く。金属ではフェルミレベルを表示し、絶縁体・半導体であれば価電子帯上端のエネルギーレベルを表示します。
-width=SIZE	図の幅のデフォルト値は 1 であるが、その値を変更したい場合はこのオプションを使う。たとえば、0.8 に変更したい場合は -width=0.8 とします。
-font=SIZE	フォントのサイズを変更したいときには、これを設定する。既定値は 14 です。 たとえば、フォントサイズを 28 にしたいならば、 -font=28 とします。
-color	グラフをカラー表示します。
-mode={total layer atom}	total を指定すると、全状態密度図が作成されます。 layer を指定すると、層分割の局所状態密度の図が作成されます。 atom を指定すると、原子分割の局所状態密度の図が作成されます。 projected を指定すると、原子軌道分割の局所状態密度の図が作成されます。 既定値は total です。
-epsf={yes no}	ポストスクリプトファイルを作成しないときには、no を指定します。既定値は yes で 指定がなければ、ポストスクリプトファイルが作成されます。
-data={yes no}	層分割や原子分割の状態密度データから eps ファイルを作成するのではなく個別のファイルに出力するときには yes を指定します。

9.6 k 点ファイル生成ツール band_kpoint.pl

バンド構造図を描くには、対称線に沿った**k**点の列を生成し、その各**k**点での固有エネルギーを `ekcal` で計算します。`ekcal` は**k**点のデータが書き込まれたファイル `kpoint.data` を読み込み各**k**点での固有エネルギーを計算します。その**k**点のファイルの生成を支援するプログラムが `band_kpoint.pl` です。`band_kpoint.pl` の入力ファイルの記述形式は以下の様になっています。

```
dkv
b1x b2x b3x
b1y b2y b3y
b1z b2z b3z
n1 n2 n3 nd # Symbol
...
```

`dkv` が**k**点の間隔, `b1x,b1y,b1z` は逆格子ベクトル \mathbf{b}_1 の *x,y,z* 成分。逆格子ベクトル $\mathbf{b}_2, \mathbf{b}_3$ についても同様です。五行目以降に特殊**k**点とそのシンボルの指定をします。シンボルの指定は必須ではありませんが、指定がある場合、バンド構造図作成の際に利用されます。整数 n_1, n_2, n_3, n_d を用いて **k**ベクトルを

$$\mathbf{k} = \frac{n_1}{n_d} \mathbf{b}_1 + \frac{n_2}{n_d} \mathbf{b}_2 + \frac{n_3}{n_d} \mathbf{b}_3$$

のように指定します。シンボルは#の後に書いてください。面心立方格子の場合の例を示します。

```
0.02          <---- k 点の間隔
-1.0  1.0  1.0
1.0 -1.0  1.0  <---- 逆格子ベクトル
1.0  1.0 -1.0
0 1 1 2 # X    <---- n1 n2 n3 nd # Symbol
0 0 0 1 # {/Symbol G}
1 1 1 2 # L
5 2 5 8 # U
1 0 1 2 # X
```

これと同じものがディレクトリ `example` にあるので、それをコピーして `band_kpoint.pl` を実行してみましょう。

```
$ cd PHASE_INST_DIR/samples/tools/work
$ cp ../example/bandkpt_fcc_xglux.in .
$ band_kpoint.pl bandkpt_fcc_xglux.in > output
```

こうすると `kpoint.data` が生成されます。これがバンド構造計算用の**k**点のファイルです。この**k**点のファイルを入力に加えて、`ekcal` で**k**点での固有エネルギーを計算してください。

9.7 バンド構造図作成ツール band.pl

9.7.1 band.pl の実行

band.pl PHASE/0 の ekcal の出力 nfenergy.data と band_kpoint.pl の入力ファイルが band.pl の 入力になります。前節の入力例で生成した kpoint.data を入力とし、ekcal で固有エネルギー計算を行い、結果得られた固有エネルギーファイル nfenergy.data が ディレクトリ example にあります。このファイルを使ってバンド構造図を描いてみましょう。example にある nfenergy.data と bandkpt_fcc_xglux.in を work にコピーし、それらを入力として band.pl を実行します。

```
$ cp ../example/nfenergy.data
$ cp
$ band.pl nfenergy.data bandkpt_fcc_xglux.in -erange
```

こうすると、EPS ファイル band_structure.eps が生成されます。このファイルをご覧になるには、ghostview や gv などのソフトウェアが必要です。\$ ghostview

または \$ gv
\$ gv band_structure.eps

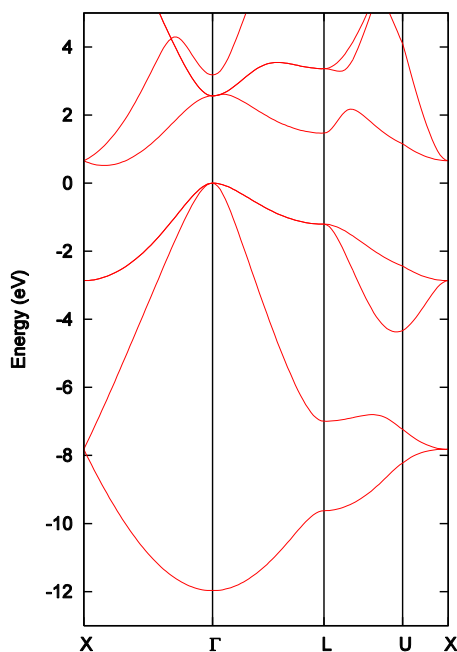


図 9-2 バルク Si のバンド構造図

band.pl を実行するときに付加した -erange は表示するエネルギーの範囲を 制御するオプション、-color はカラー出力を行うオプションです。

9.7.2 band.pl のオプション

なにも付加せずに band.pl を実行すると利用方法が表示されます。

```
$ band.pl
Usage: band.pl EnergyDataFile KpointFile -erange=Emin,Emax
-einc=dE -ptype={solid circles|lines} -with fermi
```


<code>-width=SIZE -color</code>

KpointFile の後が作図を制御するオプションです。

-erange=Emin, Emax	表示するエネルギーの範囲を eV 単位で指定する。 たとえば, -10 eV から 5 eV まで表示したい場合は, -erange=-10, 5 とします。
-einc=dE	目盛りの間隔を指定する。 たとえば 2eV 間隔に目盛りをふりたいなら, -einc=2 とします。
-ptype=TYPE	描画種を選択する。 -ptype=solid_circles : 黒く塗りつぶされた円で表示する。 -ptype=lines : 直線でつなぐ(デフォルト)。
-with_fermi	デフォルトでは描かないフェルミレベルまたは価電子帯上端のエネルギーレベルを描く。金属ではフェルミレベルを表示し, 絶縁体・半導体であれば価電子帯上端のエネルギーレベルを表示します。
-width=SIZE	図の幅のデフォルト値は 0.5 であるが, その値を変更したい場合はこのオプションを使う。たとえば, 0.3 に変更したい場合は -width=0.3 とします。
-color	グラフをカラー表示する。

9.8 原子構造の拡張 trajectory 形式への変換ツール dym2tr2.pl

Perl スクリプト `dym2tr2.pl` は、構造最適化、分子動力学法計算のデータ(`nfdym.data`)を拡張 trajectory 形式に変換します。

ツール `dym2tr2.pl` を以下のように実行します。

```
$ dym2tr2.pl nfdym.data
```

このようにすると、`dym.tr2` というファイルと `grid.mol2` というファイルが生成されます。前者は原子の座標などが記述されたファイルであり、後者是对応するセルの情報などが記述されたファイルです。

FCC のプリミティブセルに Si が二原子入った非平衡状態を初期構造とし、構造最適化した結果を拡張 trajectory 形式に変化し、可視化した例です。

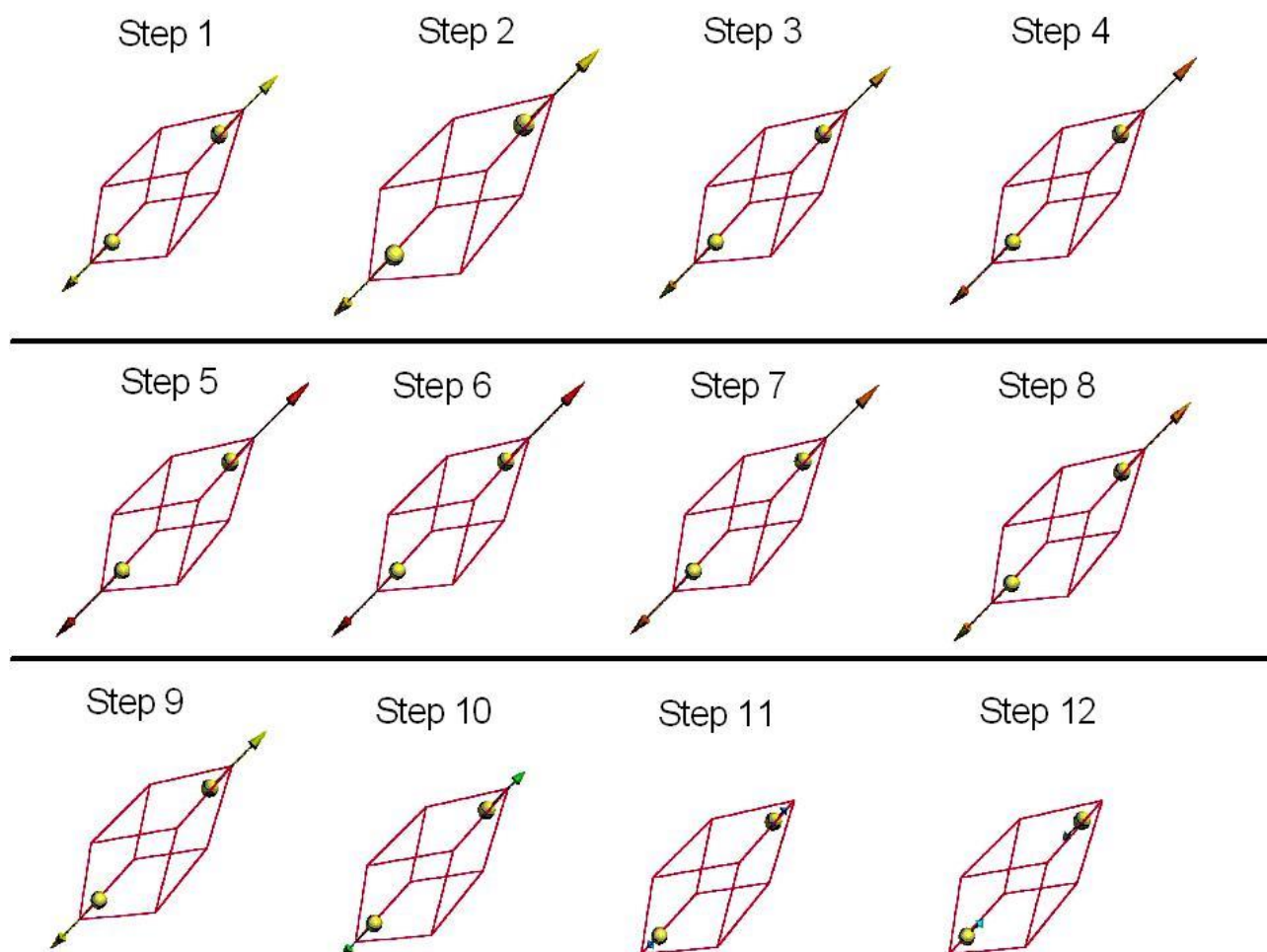


図 9-3 バルク Si の構造最適化過程の可視化例

図 9-3 の矢印は原子に作用する力を表しています。力が極大になったあとは、原子座標の更新が進む毎に原子に作用する力が小さくなり、原子構造が最適化されていく様子が分かります。図 9-3 ではプリミティブセルで表示されますが、以下のような `control.inp` というファイルを作成すれば、原点の移動やセルの変更ができます。

```
origin 1.2825 1.2825 1.2825
vector1 10.26 0 0
vector2 0 10.26 0
vector3 0 0 10.26
```

この control.inp を使用して dynm2tr2.pl で dynm.tr2 を作成すると原点が(1.2825,1.2825,1.2825) bohr に移り、セルのベクトルが(10.26,0,0), (0,10.26,0), (0,0,10.26) bohr になります。以下のようにして、dynm.tr2 を作成します。

```
$ dynm2tr2.pl nfdynm.data control.inp
```

ブラベーセルで構造最適化過程の step 10 を図示したのが、図 9-4 です。

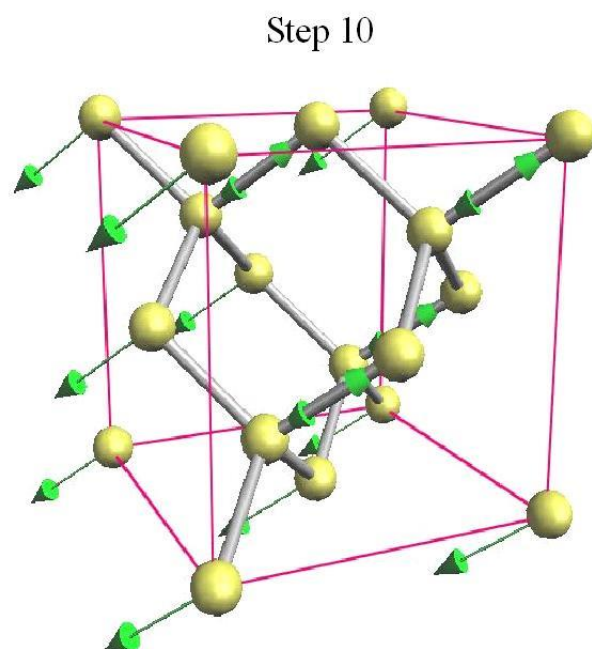


図 9-4 ブラベーセルで表したバルク Si の構造最適化過程(step 10)

9.9 振動数レベル図作成ツール freq.pl

PHASE の振動解析機能を使用すると、結晶の基準振動モードの振動数と固有ベクトルが得られます。振動解析の結果は、ファイル `mode.data` に出力されます。Perl スクリプト `freq.plmode.data` から振動数のデータを取り出し 振動数レベル図を作成します。`freq.pl` 実行すると、EPS 形式の画像ファイル `freq.eps` が出力されます。されます。

```
$ freq.pl [options] mode.data
```

バルク Si の振動数解析結果の振動数のレベル図を図 9-5 に示します。

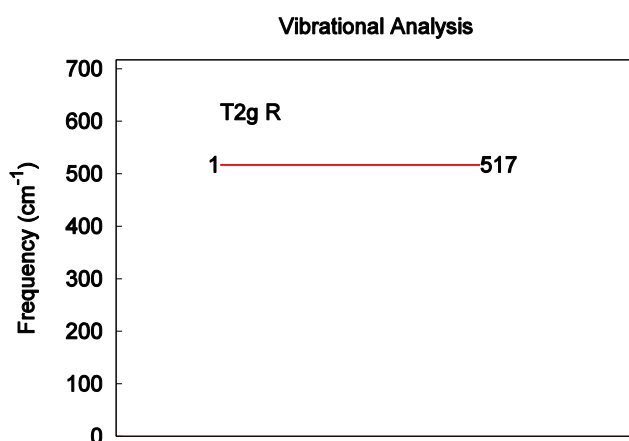


図 9-5 バルク Si の振動数レベル図

振動数レベルを表す横棒は既約表現ごとに列にまとめて分類され、その各列の上には既約表現の名称と活性を表す記号(IR,R,IR&R,NON)が表示されます。IR は赤外活性を表し、R はラマン活性を表します。IR&R は赤外活性とラマン活性があることを示します。NON はサイレントモードであることを示しています。作成された振動数レベル図では、横線の右側には振動数が cm^{-1} 単位で表示されます。既約表現ごとに振動数の低い順に番号付けされ、横線の左側に表示されます。

9.9.1 freq.pl のオプション

なにも付加せずに `freq.pl` を実行すると利用方法が表示されます。

```
$ freq.pl
```

```
*** A visualization program for vibrational frequencies ***
Usage: freq.pl [-width=W] [-height=H] [-nrep=N] {-solid|-mol|-ignored_modes=LIST}
mode.data
```

`freq.pl` のオプションです。

-width=W	図の幅のデフォルト値は 1 であるが、その値を変更したい場合はこのオプションを使う。 たとえば、0.3 に変更したい場合は -width=0.3 とします。
-height=H	図の幅のデフォルト値は 1 であるが、その値を変更したい場合はこのオプションを使う。

	たとえば, 2.5 に変更したい場合は -height=2.5 とします。
-nrep=N	一つの図に表示する既約表現の数。振動モードの既約表現がこの数よりも多いときには, 複数の EPS ファイルが作成されます。
-solid	固体の場合に並進を非表示にするオプション。 これはデフォルトで設定されています。
-mol	分子の場合に回転と並進を非表示にするオプション。
-ignored_modes=LIST	LIST のところにコンマで区切って並べた番号のモードは表示されなくなります。 たとえば, -ignored_modes=1,2,3 とすると 1,2,3 番のモードは表示されなくなります。

9.10 基準振動の軌跡の拡張 trajectory 形式ファイル変換ツール animate.pl

Perl スクリプト `animate.pl` `mode.data` に出力されている振動モードの固有ベクトルのデータを読み込み、基準振動の軌跡を拡張 trajectory 形式ファイルに変換します。

`control.inp` というファイルを用意すると、原点の移動とセルベクトルの変更ができます。

`control.inp` の例です。

```
origin 1.27189 1.27189 1.27189
vector1 10.17512 0 0
vector2 0 10.17512 0
vector3 0 0 10.17512
```

この例では、ブラベーセルで表示するために、原点を(1.27189, 1.27189, 1.27189) bohr に移し、セルベクトルを(10.17512, 0, 0), (0, 10.17512, 0), (0, 0, 10.17512) bohr に変更します。

`animate.pl` を以下のように実行します。

```
$ animate.pl mode.data control.inp
```

各振動モードごとの拡張 trajectory 形式ファイル `mode_1.tr2`, `mode_2.tr2`, ..., `mode_6.tr2` というファイルと `grid.mol2` というファイルが作成されます。拡張 trajectory 形式のファイルは振動モードの数だけ出力されます。

バルク Si の振動解析の 6 番目の基準振動の固有ベクトル `mode_6.tr2` を可視化した図を図 9-6 に示します。

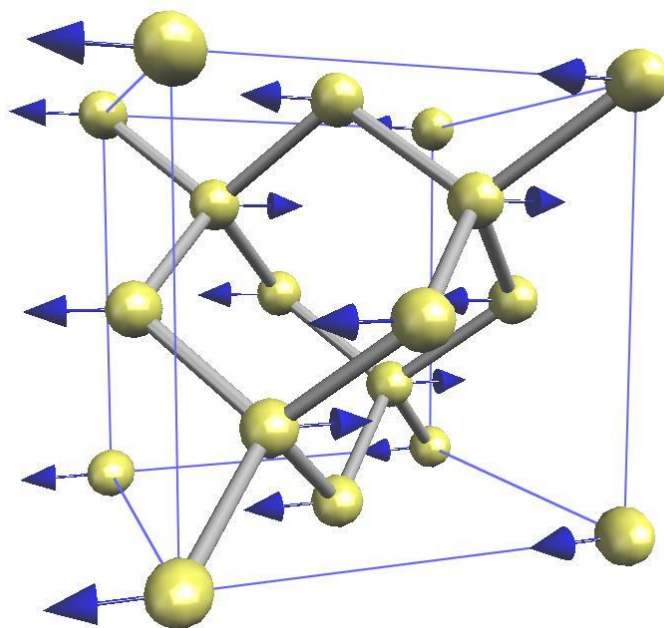


図 9-6 バルク Si の基準振動の固有ベクトル

9.11 座標データ変換ツール conv.py

conv.py という Python スクリプトを使って、座標データを変換することができます。conv.py は対話的に利用します。たとえば、nfdynm.data ファイルを CIF に変換する手続きは下記の通りです。

画面に現れる文字列	説明
\$ conv.py	
atomic configuration converter utility. Copyright (C) the RISS project, The University of Tokyo select the type of the input atomic coordinate file 0. phase_input 1. phase_output 2. VASP_input 3. VASP_output 4. OpenMX_input 5. OpenMX_output 6. XSF 7. xyz 8. cube 9. cif 10. dmol 11. LAMMPS_output x. Exit Please enter a selection (0/1/2/3/4/5/6/7/8/9/10/11/x) [0]:	変換元のファイル形式を指定する。nfdynm.data の場合 phase_output なので 1 を指定し、Enter
Please enter the name of the input atomic coordinate file, or type x to exit. [nfdynm.data]:	nfdynm.data ファイルのファイル名を指定。nfdynm.data でよいならそのまま Enter
Please enter the frame no. (enter a negative value in order to output all frames when possible), or type x to exit. [-1]:	フレームを選択する。負の値の場合「全フレーム」を選択することに相当する。カンマ区切りで三つの整数を指定することによって、始フレーム、終フレーム、間隔を指定することができる。フレームの数値は 0 始まり
select the type of the output atomic coordinate file 0. phase_input 1. phase_output 2. VASP_input 3. OpenMX_input 4. XSF 5. xyz 6. cube 7. cif 8. dmol 9. LAMMPS_input x. Exit Please enter a selection (0/1/2/3/4/5/6/7/8/9/x) [1]:	変換先の形式を指定する。CIF の場合 7 と入力し Enter
Please enter the name the output atomic coordinate file, or type x to exit. [coord.cif]:	出力ファイル名を指定する。CIF の場合、デフォルト値は coord.cif

以上の操作によって、nfdynm.data ファイルから CIF を得ることができます。そのほかのファイル形式についても同様に変換することができます。

conv.py 起動時に、以下のオプションを指定することができます。

オプション	説明
-------	----

--pack	単位胞の中に原子を押し込めます
--na=NA	<i>a</i> 軸を NA 倍にしたスーパーセルを作成し, 変換します
--nb=NB	<i>b</i> 軸を NB 倍にしたスーパーセルを作成し, 変換します
--nc=NC	<i>c</i> 軸を NC 倍にしたスーパーセルを作成し, 変換します

10. 入出力ファイル

10.1 入力ファイル

10.1.1 入力パラメータファイル nfinp.data

10.1.2 擬ポテンシャルファイル

擬ポテンシャルファイルのフォーマットについて説明します。

例として、Si 原子の擬ポテンシャルの最初の部分を以下に示します。

```
14 4 3 0 2 : zatom, ival, iloc, itpcc
ldapw91 : name
2.160000 0.860000 1.605400 -0.605400 : alp, cc
1501 96.000000 60.000000 : nmesh, xh, rmax
VALL
-0.14250064037552332E+07 -0.14102392478975291E+07 -0.13956251181755565E+07
-0.13811624288404209E+07 -0.13668496105922471E+07 -0.13526851103651347E+07
-0.13386673911985729E+07 -0.13247949320589846E+07 -0.13110662276746516E+07
-0.12974797883723934E+07 -0.12840341399159116E+07 -0.12707278233458301E+07
-0.12575593948213934E+07 -0.12445274254637859E+07 -0.12316305012010917E+07
-0.12188672226148657E+07 -0.12062362047882713E+07 -0.11937360771558125E+07
-0.11813654833546225E+07 -0.11691230810772763E+07 -0.11570075419261454E+07
-0.11450175512692606E+07 -0.11331518080976552E+07 -0.11214090248841981E+07
-0.11097879274438950E+07 -0.10982872547956155E+07 -0.10869057590252746E+07
-0.10756422051504281E+07 -0.10644953709862572E+07 -0.10534640470129563E+07
-0.10425470362444966E+07 -0.10317431540987322E+07 -0.10210512282688706E+07
-0.10104700985962711E+07 -0.99999861694454885E+06 -0.98963564707499891E+06
.....
.....
.....
```

擬ポテンシャルを格納したファイルの最初の複数の連続した行には、# で始まるコメント文を記入することができます。もしコメント文を書き入れると、PHASE を走らせたときに、標準出力 (output000) に、そのコメント文が出力されます。

プログラム PHASE に擬ポテンシャルデータを読み込ませるには、その最初の 4 行 (コメント文がある場合には、コメント文以降の 4 行目まで) に、以下のパラメーターの値が指定されている必要があります。

1 行目 natomn, ival, iloc, itpcc, igncpp

これらの変数は、それぞれ、原子番号 Z 、価電子の数 Z_v 、局在軌道の方位量子数 l_{loc} に 1 を加えた値、コアチャージ補正の有(=1)無(=0)、擬ポテンシャルデータの形式 GNCPP1(=1), GNCPP2(=2)の指定に使われます。

2 行目 xctype

交換相関相互エネルギーの型を指定します。選択できるのは、LDAPW91, GGAPBE の何れかです。

3 行目 alp1, alp2, cc1, cc2

これらのパラメーターを $\alpha_1, \alpha_2, c_1, c_2$ と書くと、PHASE の中では、コア部分の擬ポテンシャルを

$$V_{core} = -\frac{Z_v}{r} \{c_1 \operatorname{erf}(\sqrt{\alpha_1} r) + c_2 \operatorname{erf}(\sqrt{\alpha_2} r)\}$$

という式で近似して計算します。ただし、 $\text{erf}(\cdot)$ はガウスの誤差関数です。また、2つの係数 c_1 と c_2 の間には $c_1 + c_2 = 1$ の関係があります。

4行目 nmesh, xh, rmax

動径方向のメッシュを

$$r_i = r_{\max} \exp((i - N_{\text{mesh}})/x_h) \quad (i = 1, \dots, N_{\text{mesh}})$$

の式にしたがって生成します。ただし、 N_{mesh} は動径方向のメッシュの数を表します。

価電子 4 個を持つ原子番号 14 の Si 原子の、LDAPW91 法による擬ポテンシャルであることが、これらの行から分かります。

5行目(コメント文がある場合には6行目)に書かれている VALL というのは、PHASE のプログラム内で擬ポテンシャルのチェック用に使われる記号です。

その次の行からが擬ポテンシャルの実際のデータです。このデータの最初のブロックは、遮蔽された全電子ポテンシャル (screened All Electron potential, $V_{\text{scr}}^{\text{AE}}(r)$) に関するもので、そのデータ形式は、

```
do ir = 1, nmesh
   $V_{\text{scr}}^{\text{AE}}(ir)$ 
end do
```

という形になっています。

第2のブロックは、遮蔽された局所ポテンシャル (screened local potential, $V_{\text{scr},l_{\text{loc}}}^{\text{PP}}(r,l)$) に関するものです。

$V_{\text{scr}}^{\text{AE}}(r)$ 同様、そのデータ形式は、

```
do ir = 1, nmesh
   $V_{\text{scr},l_{\text{loc}}}^{\text{PP}}(ir, iloc)$ 
end do
```

となります。

第3のブロックは、価電子の電荷密度 (valence charge density, $n_v(r)$) に、球面の面積 $4\pi r^2$ をかけたものです。これを $\rho_v(r)$ とすると ($\rho_v(r) = 4\pi r^2 n_v(r)$)、そのデータは、

```
do ir = 1, nmesh
   $\rho_v(r)$ 
end do
```

と書かれています。

これらの3ブロックの記述が終った後に、軌道別に擬波動関数と擬ポテンシャルのデータが出力されます。その形式は、ノルム保存の場合とウルトラソフトの場合で全く異なります。

詳しくは、CIAO のユーザーマニュアルをご参照ください。

10.2 入出力ファイル設定ファイル file_name.data

10.3 入力ファイル（プログラム ekcal）

10.3.1 サンプリング k 点ファイル kpoint.data（F_KPOINT）

主として, ekcal によるバンド計算を行う際に利用するファイルです. 計算すべき k 点の情報が記述してあります. k 点サンプリングの方法として“file”を指定した場合必須のファイルとなります. ただし, このファイルは 通常は band_kpoint.pl スクリプト(第?節)などから作成します. 直接エディターなどで 作成する機会は多くないと思いますが, 以下参考のため記述します.

F_KPOINT ファイルは, 典型的には次のようになります.

```
141 141      (a)
0 50 50 100 1 (b)
0 49 49 100 1
0 48 48 100 1
0 47 47 100 1
0 46 46 100 1
0 45 45 100 1
0 44 44 100 1
0 43 43 100 1
.....
.....
.....
```

(a) k 点の個数を指定します. 今の場合, 141 個となります.

(b) 4 つの整数が並んでいますが, それぞれ k 点を(9)式のように定義した場合の n_1, n_2, n_3, n_d, w になります(ここで $\vec{b}_1, \vec{b}_2, \vec{b}_3$ は逆格子ベクトルです).

$$\vec{k} = w \times \left(\frac{n_1}{n_d} \vec{b}_1 + \frac{n_2}{n_d} \vec{b}_2 + \frac{n_3}{n_d} \vec{b}_3 \right)$$

10.4 出力ファイル

10.4.1 状態密度ファイル dos.data (F_DOS)

F_DOS 識別子によって指定されるファイルには、状態密度の計算結果が記入されます。既定のファイル名は dos.data です。

ファイルフォーマットとしては、まず、スピンを考慮していない計算では全状態密度のデータが次のように記述されます。

No.	E (hr.)	dos (hr.)	E (eV)	dos (eV)	sum
6	-0.20528	0.0000000000	-11.949000	0.0000000000	0.0000000000
16	-0.20491	0.0000000000	-11.939000	0.0000000000	0.0000000000
26	-0.20455	0.0000000000	-11.929000	0.0000000000	0.0000000000
.....					
.....					
.....					
END					

ここで No.の列は状態に割り振られた番号, E(hr.)はハートリー単位のエネルギー, dos(hr.)はハートリー単位でエネルギーを表した場合の 状態密度, E(eV)は電子ボルト単位でのエネルギー, dos(eV)は電子ボルト単位でエネルギーを表した場合の状態密度, sum は積算状態密度にそれぞれ対応します。他方, スピンを考慮した計算の場合以下ようになります。

No.	E (hr.)	dos_up (hr.)	dos_down (hr.)	E (eV)	dos_up (eV)	dos_down (eV)
	sum_up	sum_down	sum_total			
1	-1.5451	0.0000000000	0.0000000000	-45.4403	0.0000000000	0.00000000
000	0.0000	0.0000	0.0000			
11	-1.5441	0.0000000000	0.0000000000	-45.4131	0.0000000000	0.00000000
000	0.0000	0.0000	0.0000			
21	-1.5431	0.0000000000	0.0000000000	-45.3859	0.0000000000	0.00000000
000	0.0000	0.0000	0.0000			
31	-1.5421	0.0000000000	0.0000000000	-45.3587	0.0000000000	0.00000000
000	0.0000	0.0000	0.0000			
41	-1.5411	0.0000000000	0.0000000000	-45.3315	0.0000000000	0.00000000
000	0.0000	0.0000	0.0000			
51	-1.5401	0.0000000000	0.0000000000	-45.3043	0.0000000000	0.00000000
000	0.0000	0.0000	0.0000			

dos_up, dos_down はそれぞれアップスピンとダウンスピンの状態密度, sum_up と sum_down はそれぞれアップスピンとダウンスピンの 積算状態密度に相当します。sum_total はsum_up と sum_down の和です。

原子分割局所状態密度, 層分割局所状態密度を計算した場合, さらにこの後にどのような状態密度かを表す識別用の行の後に対応するデータが記述されます。なお, スピンを考慮した計算としない計算の違いは全状態密度の場合と同様なので以後省略します。

● 原子分割状態密度

原子分割状態密度の場合, 次のような記述が得られます。

ALDOS	num_atom =		1			
No.	E (hr.)	dos (hr.)	E (eV)	dos (eV)	sum	
6	-0.84950	0.0000000000	-26.189850	0.0000000000	0.00000000	
000						
16	-0.84850	0.0000000002	-26.162639	0.0000000000	0.00000000	
000						
					
					
					
END						
ALDOS	num_atom =		2			
					
					
					

原子分割状態密度はALDOS という文字列から始まる行以降から END 行まで記述されます. ALDOS の次に記述されている, num_atoms = 1 などの情報は, 対応する原子のインデックスです. このインデックスは入力ファイルにて指定した原子の順番と同じとなります.

- 層分割局所状態密度

層分割局所状態密度の場合次のような記述が得られます.

LAYERDOS	num_layer =		1		
No.	E (hr.)	dos (hr.)	E (eV)	dos (eV)	sum
6	-0.84950	0.0000000000	-26.189850	0.0000000000	0.0000000
000					
16	-0.84850	0.0000000002	-26.162639	0.0000000000	0.0000000
000					
				
				
				
END					
LAYERDOS	num_layer =		2		
				
				
				

基本的には原子分割局所状態密度と同等ですが, 識別子名が LAYERDOS となっていること, num_layer で入力ファイルにて指定した層番号 が記述されること, などの違いがあります.

10.4.2 エネルギー履歴ファイル nfefn.data (F_ENF)

F_ENF 識別子によって指定されるファイルには、系の全エネルギーや原子に働く力の最大値、さらに分子動力学シミュレーションを行った場合はイオンの運動エネルギーや保存量なども記述されます。構造緩和を行った場合と分子動力学シミュレーションを行った場合とで 出力内容が異なるので、それぞれについて説明します。

● 構造緩和計算

典型的な構造緩和を行った後の F_ENF ファイルの例を示します。

iter_ion	iter_total	etotal	forcmx
1	24	-108.4397629733	0.0086160410
2	40	-108.4401764388	0.0076051917
3	56	-108.4405310817	0.0068758156
4	73	-108.4410640011	0.0065717365
5	94	-108.4414256084	0.0099533097
6	113	-108.4414317178	0.0094159378
		
		
		

各列は各々次のような量に対応します。

iter_ion	イオンの更新回数です。
iter_total	SCF ループの更新回数です。この数字は通算の値が記述されます。
etotal	全エネルギーを、ハートリー単位で出力します。
forcmx	原子に働く力の最大値を原子単位(hartree/bohr ³) で記述します。この値が入力ファイルにて与えた構造緩和の収束判定を満たすまで計算は実行されます。

● 分子動力学法計算

分子動力学法計算の場合、下記のようになります。

iter_ion	iter_total	etotal	ekina	econst	forcmx
1	18	-7.8953179624	0.0000000000	-7.8953179624	0.0186964345
2	30	-7.8953851218	0.0000665502	-7.8953185716	0.0183575425
3	43	-7.8955768901	0.0002565396	-7.8953203505	0.0173392067
				
				
				

構造緩和の場合とほぼ同様ですが、新たな列が追加されます。

ekina	系の運動エネルギー、
econst	系の保存量、すなわちエネルギー一定の分子動力学シミュレーションの場合系の全エネルギー、温度一定の分子動力学シミュレーションの場合系の全エネルギーに熱浴のエネルギーを加えた量です。

● 格子最適化計算

格子の最適化を行った場合、下記のようになります。

iter_unitcell	iter_ion	iter_total	etotal	forcmx	stressmx
1	1	25	-181.4043211381	0.0019960638	
1	2	33	-181.4043560304	0.0004826299	
1	3	40	-181.4043582176	0.0000016495	0.0002327496
2	1	49	-181.4044223602	0.0000572790	0.0002273231
3	1	58	-181.4044833189	0.0001158383	0.0002220365
				
				
				

通常の構造最適化のケースに加え、以下の列が加えられます。

iter_unitcell	格子の更新回数
stressmx	ストレステンソルの最大値

10.4.3 原子座標履歴ファイル nfdynm.data (F_DYNM)

F_DYNM 識別子によって指定されるファイルには、各原子の座標とそれに働く力が記述されます。構造緩和や分子動力学シミュレーションを行った場合はイオンの更新の回数分だけデータが書き込まれます。典型的な F_DYNM ファイルの中身を以下に記述します。なお、このファイルにおいて利用される単位系はすべて原子単位系です。

```
#
# a_vector =          9.2863024980          0.0000000000          0.0000000000
# b_vector =         -4.6431512490          8.0421738710          0.0000000000      (a)
# c_vector =          0.0000000000          0.0000000000         10.2158587136
# ntyp =          2 natm =          9      (b)
# (natm->type)      1      1      1      1      1      2      2      2      (c)
# (speciesname)    1 :      O      (d)
# (speciesname)    2 :      Si
#
cps and forc at (iter_ion, iter_total =      1      24 )      (e)
  1      3.161057370      1.169332082      1.214972077      -0.004058      -0.005565      -0.004966      (f)
  2      6.693102525      2.152889944      4.620258315          0.006945      -0.001028      -0.004994
  3      4.075293851      4.719951845      8.025544553      -0.002872          0.006394      -0.004796
  4     -1.482093879      6.872841789      5.595600399      -0.004362          0.005502          0.004993
  5     -0.567857398      3.322222026      9.000886637      -0.002792      -0.006296          0.004965
  6      2.049951276      5.889283925      2.190314161          0.006974          0.000708          0.004795
  7      4.921740324      0.000000000      3.405282833          0.001436          0.000122          0.000068
  8     -2.460870162      4.262352150      6.810569070      -0.000612          0.001305      -0.000066
  9      2.182281087      3.779821719     10.215855308      -0.000660      -0.001143          0.000001
cps and forc at (iter_ion, iter_total =      2      40 )
  1      3.156999743      1.163767576      1.210005993      -0.002904      -0.005755      -0.003892
  2      6.700048015      2.151861938      4.615264365          0.006567          0.000186      -0.003832
  3      4.072421499      4.726345880      8.020748072      -0.003503          0.005487      -0.003829
  4     -1.486455954      6.878343743      5.600593135      -0.003122          0.005780          0.003831
  5     -0.570648922      3.315925959      9.005851266      -0.003532      -0.005392          0.003892
  6      2.056925355      5.889992076      2.195109289          0.006503      -0.000290          0.003828
  7      4.923176344      0.000121757      3.405351146          0.000397      -0.000013          0.000018
  8     -2.461482612      4.263656762      6.810503226      -0.000210          0.000337      -0.000017
  9      2.181621403      3.778679157     10.215856638      -0.000197      -0.000341          0.000000
      .....
      .....
      .....
      .....
      .....
```

- (a) セルベクトルが書かれています。a_vector, b_vector, c_vector にそれぞれ a 軸, b 軸, c 軸のベクトルが記述されています。
- (b) ntyp = の後には使用されている原子種の数記述されています。この例では 2 です。また、natm = の後には原子数が書かれています。この例では 9 です。
- (c) (natom→type) の後には、原子と原子種のマッピングが書かれています。この例だと、1 番目から 6 番目の原子の原子種は 1, 7 番目から 9 番目の原子種は 2 という元素に対応します。
- (d) (speciesname) の後には、原子種と ID のマッピングが書かれています。この例では、1 という原子種は O(酸素), 2 という原子種は Si(珪素) に相当する、ということになります。
- (e) 各ステップでの情報が記述されています。この例では、イオンの更新回数が 1 回、SCF の更新回数が 24 回となります。
- (f) 実際の原子の場所とその原子に働いている力が記述されています。1 番目の列は原子の ID, 2 番目から 4 番目の列が原子の場所の x,y,z 座標, 5 番目から 6 番目の列が原子に働く力の x,y,z 座標となります。もし、入力ファイルにおいて printlevel ブロックの velocity 変数を 2 に設定していた場合、7 番目から 9 番目の列に速度が原子単位で出力されます。

10.4.4 電荷密度ファイル nfchr.cube (F_CHR)

F_CHR ファイルには実空間における電荷密度のデータが記述されます。ただし、ここでは PHASE のデフォルトの出力ではなく、file_type として cube をご指定いただいた場合に得られる Gaussian Cube 形式のファイルについて説明します。PHASE デフォルトの出力を可視化することはできません。なるべく Gaussian Cube 形式のファイルをご利用いただくことをお勧めします。

This is a title line for the bulk Si	(a)
SCF Total Density	
8 0.0000 0.0000 0.0000	(b)
20 0.513000 0.000000 0.000000	(c)
20 0.000000 0.513000 0.000000	
20 0.000000 0.000000 0.513000	
14 4.000000 1.282500 1.282500 1.282500	(d)
14 4.000000 8.977500 8.977500 8.977500	
14 4.000000 1.282500 6.412500 6.412500	
14 4.000000 8.977500 3.847500 3.847500	
14 4.000000 6.412500 1.282500 6.412500	
14 4.000000 3.847500 8.977500 3.847500	
14 4.000000 6.412500 6.412500 1.282500	
14 4.000000 3.847500 3.847500 8.977500	
0.87897E-01 0.80457E-01 0.63811E-01 0.47743E-01 0.35993E-01 0.26628E-01	(e)
0.18342E-01 0.12084E-01 0.83725E-02 0.65941E-02 0.60774E-02 0.65941E-02	
0.83725E-02 0.12084E-01 0.18342E-01 0.26628E-01 0.35993E-01 0.47743E-01	
0.63811E-01 0.80457E-01 0.80457E-01 0.76575E-01 0.63379E-01 0.51118E-01	
0.43367E-01 0.35993E-01 0.26413E-01 0.17302E-01 0.11265E-01 0.80672E-02	
0.65941E-02 0.62411E-02 0.68963E-02 0.88010E-02 0.12493E-01 0.18342E-01	
0.26413E-01 0.37600E-01 0.53180E-01 0.70418E-01 0.63811E-01 0.63379E-01	
.....	
.....	
.....	
.....	
.....	

- (a) タイトルやコメント領域です。一行目は入力ファイルからご指定いただくこともできます。
- (b) 8 は原子の個数, “0.0000 0.0000 0.0000” は原点です。原点は, PHASE では常に (0,0,0) です。
- (c) セルの情報が与えられています。たとえば, “20 0.513000 0.000000 0.000000” とある場合, 一つ目の軸を 20 分割し, 一分割の単位は 0.513, 0.00, 0.00 であることを意味します。長さの単位は Bohr です。
- (d) 一つ目の数字は原子番号です。今の例では 14 なので, シリコンであることが分かります。次の 4.00000 という数字は, 価電子数です。その隣の三つの数字は対応する原子の x,y,z 座標です。単位は Bohr です。
- (e) 実際の電荷密度の情報が書き出されています。まず z 座標, 次に y 座標, 最後に x 座標が変化するような順で出力されます。
 (1,1,1) (1,1,2) (1,1,20) (1,2,1) (1,2,2)
 (1,20,20) (2,1,1)
 (20,20,19) (20,20,20)

10.4.5 継続計算用ファイル continue.data (F_CNTN)

このファイルには、変更する可能性のある継続計算用のデータが記述されています。たとえば、継続計算を行う際に電子状態の収束判定を変更したい場合や、すでに収束した計算を再度継続して計算したい場合などはこのファイルを編集する必要があります。その内容は、たとえば以下ようになります。

iteration, iteration_ionic, iteration_electronic			
19 1 19			(a)
Ionic System			
(natm)			
2			(b)
(pos)			
0.1249999999999999D+00 0.1250000000000001D+00 0.1250000000000001D+00			(c)
0.8749999999999994D+00 0.8749999999999994D+00 0.8749999999999994D+00			
(cps)			
0.1282864712563094D+01 0.1282864712563093D+01 0.1282864712563093D+01			(d)
0.8980052987941646D+01 0.8980052987941646D+01 0.8980052987941646D+01			
(cpd)			
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00			
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00			
(cpo(1))			
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00			
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00			
(cpo(2))			
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00			
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00			
(cpo(3))			
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00			
0.0000000000000000D+00 0.0000000000000000D+00 0.0000000000000000D+00			
Total Energy			
-0.7851066208137508D+01 -0.7851066208137508D+01			(e)
isolver			
17			
convergence			
2			(f)
edelta_ontheway			
0.1000000000000000D-07			(g)

代表的な項目について説明します。

- (a) 積算した更新回数, イオンの更新回数, SCF 計算の更新回数が出力されます。
- (b) 原子の数が出力されます。
- (c) 原子の座標が, セルベクトルに対する値で出力されます。
- (d) 原子の座標が, 直交座標系で, bohr 単位で出力されます。
- (e) 一つ前のステップと現在のステップの全エネルギーが出力されます。
- (f) 収束状況が出力されます。
0: 未収束, 1: SCF は収束しているがイオンの更新は未収束, 2: 収束済み
を意味します。特に 2 の場合で継続計算を行うと, 計算開始と同時にポスト処理に入ります。「一旦は収束したものの, 条件を変更した後に継続計算を行いたい」などの状況においては, この値を 0 としてください。
- (g) SCF の収束判定の値が出力されます。計算途中にて SCF 計算の収束判定を変更する場合, 入力ファイルだけではなくこちらでも変更してください。

10.4.6 固有値データファイル nfenergy.data (F_ENERG)

ekcal による固有値計算の結果が書き込まれるファイルです。典型的な例を以下に記します。

```

num_kpoints =      117                                (a)
num_bands   =       8                                (b)
nspin       =       1                                (c)
Valence band max = 0.233846                            (d)

nk_converged =      117                                (e)
ik = 1 ( 0.500000 0.500000 0.000000 )
ik = 2 ( 0.487805 0.487805 0.000000 )
ik = 3 ( 0.475610 0.475610 0.000000 )
ik = 4 ( 0.463415 0.463415 0.000000 )
ik = 5 ( 0.451220 0.451220 0.000000 )
ik = 6 ( 0.439024 0.439024 0.000000 )
...
...
...

=== energy_eigen_values ===
ik = 1 ( 0.000000 0.500000 0.500000 )                (f)
      -0.0484324576 -0.0484324576 0.1258094928 0.1258094928 (g)
      0.2619554301 0.2619554301 0.6015285208 0.6015285208
=== energy_eigen_values ===
ik = 2 ( 0.000000 0.490000 0.490000 )
      -0.0540717201 -0.0427149632 0.1258687739 0.1258687739
      0.2607026807 0.2633829927 0.6006243932 0.6006243932
      .....
      .....
      .....

```

- (a) k 点の数が書いてあります。この例では 117 個です。
- (b) バンドの数が記述してあります。この例では 8 です。
- (c) スピン自由度が記述してあります。1 か 2 の値をとります。この例では 1 であり、スピン分極を考慮しない計算に対応します。
- (d) フェルミエネルギーの値が書いてあります。半導体/絶縁体の場合価電子帯の上端のエネルギーが記述されます。単位はハートリーです。
- (e) 計算した k 点が記述されます。
- (f) 以下、固有値の情報が記述されます。まずこの行で、どの k 点に対応する固有値データかが分かります。この例では、1 番目の k で、その座標は逆格子ベクトルを基底として (0,0.5,0.5) となります。
- (g) 固有値のデータが、バンドの数だけ出力されます。単位はハートリーです。

スピンを考慮した計算の場合(上記の(c)が 2 の場合)もほぼ同様のファイル形式ですが、上記の(e)の隣に“UP”か“DOWN”と記述される、という違いがあります。それぞれ多数派スピンと少数派スピンに対応する固有値が書き出されます。

```

      .....
      .....
      .....

=== energy_eigen_values ===
ik = 1 ( 0.000000 0.000000 0.000000) UP
      -0.1998699758 0.0267639589 0.0267639589 0.0267639589
      0.0725171077 0.0725171077 1.0289118953 1.0289118953
      1.0289118953 1.1650173104 1.1650173104 1.1650173104
      1.2129026022 1.2129026022 1.2994754011 1.2994754011
      1.2994754011 1.6365336765 2.2629596795 2.2629596795
=== energy_eigen_values ===
ik = 2 ( 0.000000 0.000000 0.000000) DOWN
      -0.1960420390 0.1062941746 0.1062941746 0.1062941746
      0.1799862148 0.1799862148 1.0183970612 1.0183970612
      1.0183970612 1.2174266166 1.2174266166 1.2192701193

```

1.2192701193	1.2192701193	1.3289165100	1.3289165100
1.3289165100	1.6910264603	2.2876818717	2.2876818717
		
		
		

11. 誘電応答解析機能 UVSOR

11.1 はじめに

11.1.1 UVSOR とは

UVSOR(Universal Virtual Spectroscope for Optoelectronics Research)は第一原理擬ポテンシャル法に基づいて、物質の誘電・光学応答を原子レベルで計算する機能の総称です。2つのプログラム `epsmain` と `tdlrmain` が含まれます。またプログラム `phase` および `ekcal` にも誘電応答解析機能 UVSOR が含まれています。UVSOR は、プログラム `phase` での SCF 計算結果をもとに、物質の誘電関数を 0Hz～紫外波長域の周波数で計算することができます。誘電関数は、物質の電氣的・光学的性質を決定する重要な物理量で、その測定はエレクトロニクス及び光学における主要な課題です。誘電関数は分光器を用いて測定することができますが、現実の分光器は測定周波数域が限られているので、実験的に誘電関数を決定するには、いろいろな種類の分光器を使い分けます。たとえば、エレクトロニクス分野で興味がある 10^6 (Mega) 10^9 (Giga)Hz 域での誘電関数を測定するには、その周波数域の電磁波を発生させる RF(Radio Frequency)発振器を使用します。また、光学において興味がある赤外、可視、紫外波長域における誘電関数を測定するには、それぞれの波長域専用の分光器を使用します。

しかし、発振器および分光器の調整はかならずしも容易ではなく、このことが誘電関数の測定を困難にしています。また、 10^{12} (Tera)Hz 域の分光器は、現在良好に動作するものが存在しません。このため、この波長域での誘電関数測定は極めて困難であるのが現状です。

UVSOR はこのような問題を解決する材料シミュレーション・ソフトウェアです。解析可能な波長域が、0Hz～紫外波長域の周波数と極めて広く、エレクトロニクス・光学の分野において興味あるほとんど全ての波長域における誘電関数を第一原理法に基づいて計算することができます。UVSOR は、その名が示すように、一種の"万能仮想分光器"として作用し、電子・光学材料の開発に用いることができます。

図 11-1 は、物質の誘電関数（実部）の周波数依存性を模式的に示しています。

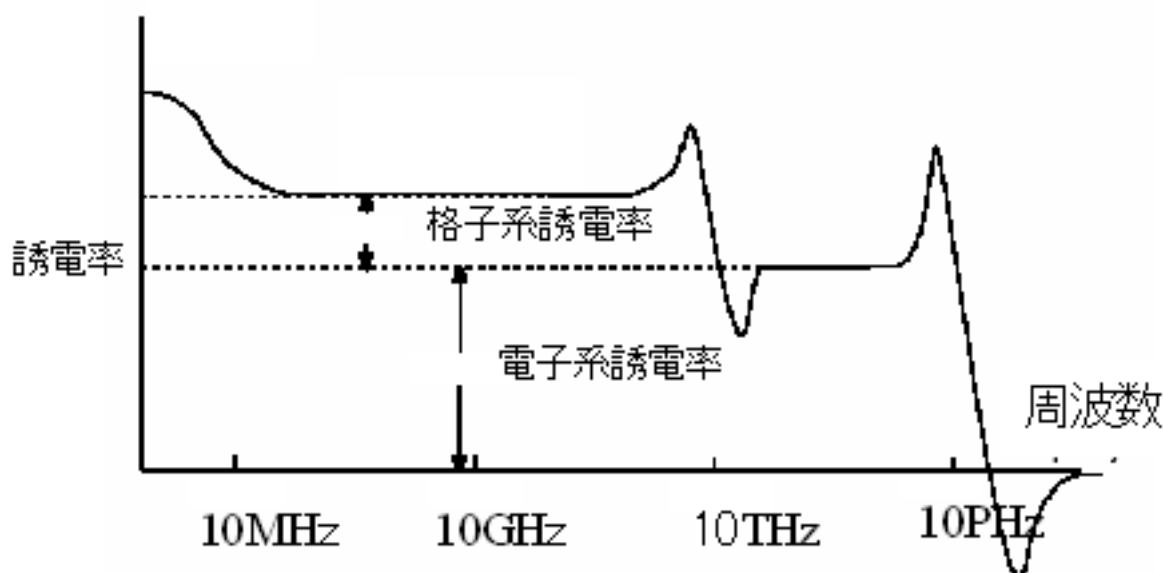


図 11-1 誘電率の誘電分散

物質の誘電率は、近似的に、物質の電子状態に起因する電子誘電率と、格子振動状態に起因する格子系誘電率の和で与えられ、電磁波の周波数が格子振動の周波数よりも低い場合の 10T(Tera)Hz 以下の RF 領域では、全誘電率は電子系及び格子系誘電率の和で与えられます。一方、電磁波の周波数が格子振動の周波数よりも高い場合の 10THz 以上の光学域では、電子系誘電率のみが全誘電率に寄与します。

誘電率計算には、電子系の応答を解析する UVSOR-Epsilon の計算体系と、格子系の応答を解析する UVSOR-Berry-Phonon の計算体系があり、これらの計算体系に従えば、はそれぞれ電子系及び格子系誘電関数を計算することができます。これにより、静的～光学波長域 (0Hz～紫外域) に及ぶ極めて広い周波数帯域での誘電関数を得ることができます。UVSOR は、誘電関数、非線形光学感受率 (2 次及び 3 次)、ならびに電子・正孔の有効質量を計算することができます。UVSOR は一種の仮想分光器として作用し、誘電体材料ならびに光学・レーザ材料の設計に用いることができます。誘電応答解析機能 UVSOR は、高速化・MPI 並列化されています。

11.1.2 UVSOR の特徴と機能

UVSOR-Epsilon はプログラム PHASE の SCF 計算結果に基づき、電子バンド構造を計算し、電子系誘電関数の計算を行うプログラムである。その特徴は以下の通りである。

UVSOR-Epsilon の特徴と機能

1. PHASE/EKCAL と入力ファイルを共用している

入力は、PHASE/EKCAL の入力ファイルに誘電関数計算用の入力タグを追加した形式となっており、容易に計算を行うことができる。

2. PHASE の計算結果を利用して誘電関数を計算する

PHASE 計算により得られる電子密度より電子バンド構造を求め、電子系誘電関数を計算する。電子バンド構造は、バンド数及び k 点数を任意に変えて誘電関数を計算でき、精度の高い誘電関数計算を行うことができる。

3. 複素誘電関数の計算

乱雑位相近似 (Random Phase Approximation) に基づき誘電関数の虚部を求め、虚部をクラマース・クロニヒ変換することにより誘電関数の実部を計算している。[1]

4. 全電子計算と同じ結果を得ることができる

擬ポテンシャルが遷移モーメントに及ぼす効果[2, 3, 4] を補正することにより、全電子計算と同じ誘電関数を得ることができる。擬ポテンシャルは、CIAO を用いて作成したノルム保存 Troullier-Martin(TM) 型ポテンシャル、及びノルム非保存ウルトラソフト型ポテンシャルを用いることができる。局所ポテンシャルは、TM 型で計算された軌道ポテンシャル、BHS、多項式ポテンシャルを扱うことができる。

遷移モーメント補正法は、Read and Needs(RN) 法[3] あるいは Kageshima-Shiraishi(KS) 法[4] のいずれかを選択できる。ノルム保存型擬ポテンシャルの場合は RN 法あるいは KS 法を、ウルトラソフト型擬ポテンシャルの場合は KS 法を用いて補正を行う。

5. 半導体・絶縁体のほか、金属、磁性材料の取り扱いができる

半導体・絶縁体のほか、金属、磁性材料に対応したリニアテトラヘドロン法[5] 及び gaussian/parabolic smearing 法を実装している。磁性材料の場合、各スピン状態の誘電関数への寄与を解析できる。金属の場合、ドルーデ項を考慮した計算を行うことが可能である。リニアテトラヘドロン計算は、UVSOR 1.00 の場合よりも約 2 倍高速化されている。

6. 光学スペクトルの計算

誘電関数より光学スペクトル (屈折率、吸収係数、反射スペクトル) を計算できる。偏光は直線偏光及び非偏光を取り扱うことができる。

7. 2 次非線形光学感受率の計算

第 2 高調波発生(Second Harmonic Generation(SHG)) 過程の 2 次非線形光学感受率 $\chi^{(2)}(-2\omega; \omega, \omega)$ 及び第 3 高調波発生 (Third Harmonic Generation(THG)) の 3 次非線形光学感受率 $\chi^{(3)}(-3\omega; \omega, \omega, \omega)$ の実部及び虚数を波長依存性を考慮して、全電子計算と同じ精度で計算できる。

8. 有効質量の計算

電子及び正孔の有効質量を計算できる。計算には kp 摂動法を用いている。

9. ユーティリティプログラム

誘電関数、非線形光学感受率計算結果を取り扱うためのユーティリティプログラムを備えている。

10. 並列計算

ソースコードは MPI 並列化されている。

UVSOR-Berry-Phonon の特徴と機能

1. PHASE/EKCAL に組み込まれている。

入力は、PHASE/EKCAL の入力ファイルに **Berry** 位相計算用および格子誘電率計算用の入力タグを追加した形式となっており、容易に計算を行うことができる。

2. ボルン有効電荷の計算

ベリー位相分極理論に基づき結晶の分極の変化を計算して、ボルン有効電荷を計算することができる。

3. 格子誘電率の計算

振動解析の結果とボルン有効電荷から格子誘電率を自動的に計算できる。

4. 圧電定数の計算

ベリー位相分極理論に基づき結晶の分極の変化を計算して、圧電定数のイオン固定項を計算することができる。振動解析の結果とボルン有効電荷とひずみ・力結合定数から圧電定数の内部ひずみ項を自動的に計算できる。

11.1.3 更新履歴

2009 年 6 月以降の更新履歴は下記の通り。

バージョン 3.20 2009/06 公開	PHASE ver 8.00 に対応
バージョン 3.21 2010/03 公開	PHASE ver 8.01 に対応
バージョン 3.30 2010/06 公開	PHASE ver 9.00 に対応 DFT+U を利用した誘電関数計算に対応
バージョン 3.40 2011/06 公開	PHASE ver 10.00 に対応
バージョン 3.41 2011/08 公開	PHASE ver 10.01 に対応 TDDFT 計算機能の実装
バージョン 3.42 2012/06 公開	PHASE ver 11.00 に対応 不具合の修正
2014/04 公開	PHASE /0 に統合

11.1.4 パッケージの構成

本プログラムは、プログラムパッケージ PHASE/0 に統合されている。

ディレクトリ			
phase0_2015.01	bin		epsmain, tdlrmain
	samples/uvsol/		本プログラムの入力例
		electron	UVSOR-Epsilon の入力例 Si 結晶の入力例 Cu 銅の入力例 AlN 窒化アルミニウムの入力例 NiO DFT+U 法を利用して誘電関数を計算する例
		lattice	UVSOR-Berry-Phonon の入力例 GaAs GaAs 結晶の入力例 AlN 窒化アルミニウムの入力例 Quartz 水晶の入力例
		lr-tddft	TDDFT の入力例 C6H6 C6H6 ₆ 分子の計算例 SiBulk Si 結晶の計算例
	util		eps_file.f90, nlo_file.f90

計算例題は、samples/uvsolr である。

samples/uvsolr/electron には UVSOR-Epsilon の入力例、samples/uvsolr/lattice には UVSOR-Berry-Phonon の入力例があります。samples/uvsolr/lr-tddft には TDDFT の入力例がある。

samples/uvsolr/electron には、Si 結晶の入力例、銅 Cu の入力例、窒化アルミニウム AlN の入力例があります。NiO は、DFT+U 法を利用して誘電関数を計算する例である。

各入力例ディレクトリ下位には、UVSOR-Epsilon の実行に必要な電子密度を phase により計算するためのディレクトリ scf、および UVSOR-Epsilon により誘電関数計算を行うためのディレクトリ eps、および擬ポテンシャルファイルを格納するディレクトリ PP が存在する。Si 入力例ディレクトリ下位には、有効質量計算を行うためのディレクトリ mass 及び 3 次非線形光学感受率の計算を行うためのディレクトリ chi3 が存在する。AlN 入力例ディレクトリ下位には、2 次非線形光学感受率計算を行うためのディレクトリ chi2_p 及び chi2_t が存在する。chi2_p は parabolic smearing 計算用、chi2_t は、リニアテトラヘドロン計算用である。

samples/uvsolr/lattice には、GaAs 結晶の入力例、窒化アルミニウム AlN の入力例、水晶 Quartz の入力例がある。各入力例ディレクトリ下位には、ベリー位相を計算するためのディレクトリ berry、および振動解析および格子誘電率計算を行うためのディレクトリ phonon、および擬ポテンシャルファイルを格納するディレクトリ PP が存在する。

samples/uvsolr/lr-tddft には、C6H6₆ 分子の計算例、Si 結晶の計算例がある。

11.2 計算手法

11.2.1 電子系

11.2.1.1 計算手法[1]

電子系誘電率は、誘電体の電子が入射電磁波の電磁場と相互作用し、価電子帯から伝導帯に遷移することによって起因する。電子系誘電率は、電磁波が引き起こす電子の遷移確率より求めることができる。本節では、以下電子の遷移確率を求め、電子系誘電率を計算する方法を説明する。

11.2.1.2 遷移確率の計算

電磁波と相互作用している誘電体結晶の1電子ハミルトニアンは(1)式で与えられる。

$$H = \frac{1}{2me}(\mathbf{p} + e\mathbf{A})^2 + V(\mathbf{r}) \quad (1)$$

m は電子の質量、 e は電荷素量、 \mathbf{p} は運動量演算子、 \mathbf{A} は電磁波のベクトルポテンシャル、 $V(\mathbf{r})$ は結晶のポテンシャルである。誘電体と電磁波の相互作用を表す1次の摂動ハミルトニアンは(2)式となる。

$$H_{int} = (e/m)\mathbf{A} \cdot \mathbf{p} \quad (2)$$

電磁波が平面波である場合、ベクトルポテンシャルは次式で与えられる。

$$\mathbf{A} = A_0 \mathbf{u} \exp [i(\mathbf{k} \cdot \mathbf{r} - \omega t)] \quad (3)$$

ここで、 \mathbf{u} は電磁波の偏光ベクトル、 \mathbf{k} は波数ベクトル、 \mathbf{r} は位置ベクトル、 ω は振動数、 t は時間である。

電磁波との相互作用により価電子帯の電子が時間 t の後に伝導帯の軌道に遷移する確率 w は、次式で与えられる。

$$w(\omega, t, \mathbf{k}_v, \mathbf{k}_c) = \frac{e^2}{m^2 \hbar^2} \left| \int_0^t dt' \int_V d\mathbf{r} \Psi_v(\mathbf{k}_v, \mathbf{r}, t') \mathbf{A}(\mathbf{k}, \mathbf{r}, t') \cdot \mathbf{p} \Psi_c(\mathbf{k}_c, \mathbf{r}, t') \right|^2 \quad (4)$$

Ψ_v は誘電体の価電子帯電子の軌道、 \mathbf{k}_v は Ψ_v の波数ベクトル、 Ψ_c は伝導帯の軌道、 \mathbf{k}_c は Ψ_c の波数ベクトルである。インデックス c 及び v はスピンインデックスを含む。 Ψ_v 及び Ψ_c は同じスピンを有する軌道である。 Ψ_v 及び Ψ_c は以下のように書き表すことができる。

$$\Psi_v(\mathbf{k}_v, \mathbf{r}, t') = \exp \left[-\frac{i}{\hbar} E_v(\mathbf{k}_v t') \right] \exp (i \mathbf{k}_v \cdot \mathbf{r}) u_v(\mathbf{k}_v, \mathbf{r}) \quad (5)$$

$$\Psi_c(\mathbf{k}_c, \mathbf{r}, t') = \exp \left[-\frac{i}{\hbar} E_c(\mathbf{k}_c t') \right] \exp (i \mathbf{k}_c \cdot \mathbf{r}) u_c(\mathbf{k}_c, \mathbf{r}) \quad (6)$$

(5)及び(6)式を(4)式に代入し、 t' に関して部分積分を行うことにより、次式を得る。

$$w(\omega, t, \mathbf{k}_v, \mathbf{k}_c) = \frac{e^2 E_0^2}{m^2 \omega_{cv}^2} \left| \int_0^t dt' \exp \left[i \hbar^{-1} (E_c(\mathbf{k}_c) - E_v(\mathbf{k}_v) - \hbar \omega) t' \right] \mathbf{u} \cdot \mathbf{M}_{vc} \right|^2 \quad (7)$$

ここで、 $\mathbf{E} = -\partial \mathbf{A} / \partial t = E_0 \mathbf{u} \exp [i(\mathbf{k} \cdot \mathbf{r} - \omega t)]$ 及び $\mathbf{p} = \frac{\hbar}{i} \nabla$ の関係式を用いた。 $\omega_{c,v}$ 及び $\mathbf{u} \cdot \mathbf{M}_{vc}$ は、それぞれ(8)及び(9)式で計算される量である。

$$\omega_{c,v} = \frac{1}{\hbar} (E_c(\mathbf{k}_c) - E_v(\mathbf{k}_v)) \quad (8)$$

$$\mathbf{u} \cdot \mathbf{M}_{vc} = \int_V d\mathbf{r} \exp [-(\mathbf{k}_c - \mathbf{k}) \cdot \mathbf{r}] u_c^* \mathbf{u} \cdot \nabla \exp (i \mathbf{k}_v \cdot \mathbf{r}) u_v \quad (9)$$

∇ は(10)式で表される微分演算子である。

$$\nabla = i \frac{\partial}{\partial x} + j \frac{\partial}{\partial y} + k \frac{\partial}{\partial z} \quad (10)$$

i, j, k はそれぞれ、 x, y, z 方向の単位ベクトルである。(7)式を積分することにより次式を得る。

$$w(\omega, t, \mathbf{k}_v, \mathbf{k}_c) = \frac{e^2 E_0^2}{m^2 \omega_{cv}^2} \left| \frac{\exp \left[\frac{i(E_c - E_v - \hbar\omega)t}{\hbar} \right] - 1}{\frac{i(E_c - E_v - \hbar\omega)t}{\hbar}} \mathbf{u} \cdot \mathbf{M}_{vc} \right|^2 \quad (11)$$

(11)式の両辺を t で微分することにより、次式を得る。

$$\frac{\partial w}{\partial t} = \frac{e^2 E_0^2}{m^2 \omega_{cv}^2} |\mathbf{u} \cdot \mathbf{M}_{vc}|^2 2\pi\hbar\delta(E_c - E_v - \hbar\omega) \quad (12)$$

(12)式は、単位時間あたりに電子が Ψ_v から Ψ_c へ遷移する確率を与える。単位体積あたりの全電子遷移確率 W_{vc} は、次式により得られる。

$$W_{vc} = \frac{e^2 E_0^2}{m^2 V} \sum_{\mathbf{k}, c, v} \frac{|\mathbf{u} \cdot \mathbf{M}_{vc}|^2}{\omega_{cv}^2} 2\pi\hbar\delta(E_c - E_v - \hbar\omega) \quad (13)$$

ここで V は誘電体の体積であり、 Σ は全ての \mathbf{k} 、価電子・伝導帯軌道の組み合わせについて和をとることを意味する。

11.2.1.3 電子系誘電率の計算

誘電体に入射された電磁波は、誘電体の電子遷移を引き起こし、エネルギーを失う。そのエネルギー損失量は $W_{vc}\hbar\omega$ である。一方、マックスウェルの理論では、その損失量は σE_0^2 となる。 σ は誘電体のオプティカルコンダクティビティである。従って、

$$W_{vc}\hbar\omega = \sigma E_0^2/2 \quad (14)$$

一方、 σ と電子系誘電率の虚部 ϵ_2 の間には次の関係式がある。

$$\epsilon_2 = 4\pi\sigma/\omega \quad (15)$$

(14)及び(15)式より ϵ_2 の計算式を得る。

$$\begin{aligned} \epsilon_2 &= \frac{8\pi e^2 \hbar^2}{m^2 V} \sum_{\mathbf{k}, c, v} \frac{|\mathbf{u} \cdot \mathbf{M}_{vc}|^2}{\omega_{cv}^2} \delta(E_c(\mathbf{k}_c) - E_v(\mathbf{k}_v) - \hbar\omega) \\ &= \frac{8\pi e^2 \hbar^4}{m^2 V} \sum_{\mathbf{k}, c, v} \frac{|\mathbf{u} \cdot \mathbf{M}_{vc}|^2}{(E_c(\mathbf{k}_c) - E_v(\mathbf{k}_v))^2} \delta(E_c(\mathbf{k}_c) - E_v(\mathbf{k}_v) - \hbar\omega) \end{aligned} \quad (16)$$

(16)式の計算を導入するために近似を導入する。電磁波の波長は、誘電体結晶のユニットセルの大きさよりもはるかに長い。従って、 $\mathbf{k}_c, \mathbf{k}_v \gg \mathbf{k}$ であるので、以下のように近似できる。

$$\mathbf{u} \cdot \mathbf{M}_{vc} \cong \int_V d\mathbf{r} \exp(-\mathbf{k}_c \cdot \mathbf{r}) u_c^* \mathbf{u} \cdot \nabla \exp(i\mathbf{k}_v \cdot \mathbf{r}) u_v \quad (17)$$

運動量演算子 \mathbf{p} を使うと

$$\mathbf{u} \cdot \mathbf{M}_{vc} \cong \frac{i}{\hbar} \langle \Psi_c(\mathbf{k}_c) | \mathbf{u} \cdot \mathbf{p} | \Psi_v(\mathbf{k}_v) \rangle \quad (18)$$

$$= \frac{i}{\hbar} \langle \Psi_c(\mathbf{k}_a) | \mathbf{u} \cdot \mathbf{p} | \Psi_v(\mathbf{k}_a) \rangle \quad (19)$$

となる。 $\mathbf{k}_c = \mathbf{k}_v = \mathbf{k}_a$ である。 $\langle \Psi_c(\mathbf{k}_c) | \mathbf{u} \cdot \mathbf{p} | \Psi_v(\mathbf{k}_v) \rangle$ の値は、 $\mathbf{k}_c = \mathbf{k}_v$ の場合のみ零でない。(16)式及び(19)式より、 \mathbf{p} 表示の ϵ_2 計算式が得られる。

$$\epsilon_2 = \frac{8\pi e^2 \hbar^2}{m e^2 V} \sum_{\mathbf{k}_a, c, v} \frac{|\langle \Psi_c(\mathbf{k}_a) | \mathbf{u} \cdot \mathbf{p} | \Psi_v(\mathbf{k}_a) \rangle|^2}{(E_c(\mathbf{k}_a) - E_v(\mathbf{k}_a))^2} \delta(E_c(\mathbf{k}_a) - E_v(\mathbf{k}_a) - \hbar\omega) \quad (20)$$

速度演算子 $\mathbf{v} = \mathbf{p}/m$ を用いると、 \mathbf{v} 表示の計算式[11]が得られる。

$$\epsilon_2 = \frac{8\pi e^2 \hbar^2}{V} \sum_{\mathbf{k}_a, c, v} \frac{|\langle \Psi_c(\mathbf{k}_a) | \mathbf{u} \cdot \mathbf{v} | \Psi_v(\mathbf{k}_a) \rangle|^2}{(E_c(\mathbf{k}_a) - E_v(\mathbf{k}_a))^2} \delta(E_c(\mathbf{k}_a) - E_v(\mathbf{k}_a) - \hbar\omega) \quad (21)$$

遷移モーメントには、次の関係式が成り立つ。

$$\langle \Psi_c(\mathbf{k}_a) | \mathbf{p} | \Psi_v(\mathbf{k}_a) \rangle = \frac{im}{\hbar} (E_c(\mathbf{k}_a) - E_v(\mathbf{k}_a)) \langle \Psi_c(\mathbf{k}_a) | \mathbf{r} | \Psi_v(\mathbf{k}_a) \rangle \quad (22)$$

(20)及び(22)式より \mathbf{r} 位置演算子 \mathbf{r} 表示の計算式を得る。

$$\epsilon_2 = \frac{8\pi e^2}{V} \sum_{\mathbf{k}_a, c, v} |\langle \Psi_c(\mathbf{k}_a) | \mathbf{u} \cdot \mathbf{r} | \Psi_v(\mathbf{k}_a) \rangle|^2 \delta(E_c(\mathbf{k}_a) - E_v(\mathbf{k}_a) - \hbar\omega) \quad (23)$$

比誘電率は、得られた ϵ_2 を真空の誘電率 ϵ_0 で割ることにより得られる。電子系誘電率の実部 ϵ_1 は ϵ_2 のクラマース・クロニツヒ変換 ((24)式) により計算される。 P はコーシーの主値を取ることを意味する

$$\epsilon_1(\omega) = 1 + \frac{2}{\pi} P \int_0^\infty \frac{\Omega \epsilon_2(\Omega)}{\Omega^2 - \omega^2} d\Omega \quad (24)$$

本プログラムは(23)式により ϵ_2 を求め、(24)式により ϵ_1 を計算する。

11.2.1.4 光学スペクトルの計算法

電子系誘電率 $\epsilon = \epsilon_1 + i\epsilon_2$ より、以下の光学スペクトルを計算することができる。

- 複素屈折率 $N = n + ik$: $N = \epsilon^{1/2}$
- 吸収係数 η : $\eta = \frac{2k\omega}{c}$
- 反射スペクトル R : $R = \frac{(n-1)^2 + k^2}{(n+1)^2 + k^2}$

11.2.1.5 第一原理法による計算法

電子系誘電率には、遷移モーメント $\langle \Psi_c | \mathbf{r} | \Psi_v \rangle$ を計算することが必要である。本節では第一原理擬ポテンシャル法による遷移モーメントの計算法について説明する。第1原理擬ポテンシャル法では、内殻電子が価電子に及ぼす効果を擬ポテンシャルに置き換え、価電子のみを扱うことにより電子状態を計算する。擬ポテンシャル法での誘電体の1電子ハミルトニアンは次式で与えられる。

$$H = \frac{1}{2m} \mathbf{p}^2 + V(\mathbf{r}, \mathbf{p}) \quad (25)$$

V は結晶のポテンシャルである。擬ポテンシャル法では、電子の運動量 \mathbf{p} に依存するノンローカルポテンシャルを用いる。電磁波と相互作用する誘電体の1電子ハミルトニアンは、次式で与えられる。

$$H = \frac{1}{2m} (\mathbf{p} + e\mathbf{A})^2 + V(\mathbf{r}, \mathbf{p} + e\mathbf{A}) \quad (26)$$

摂動ハミルトニアン H_{int} は(27)式である。

$$H_{int} = (e/m)\mathbf{A} \cdot \mathbf{p} + ppc \quad (27)$$

(27)式の右辺第1項は、(2)式の摂動ハミルトニアンである。第2項 (ppc 項) は V に由来する摂動ハミルトニアンである。第一原理法で遷移モーメントを計算する場合、ppc 項を考慮して計算を行う必要がある。第1項に起因する遷移モーメント成分をローカル項、ppc 項に起因する成分をノンローカル項あるいは ppc 項と呼ぶことにする。ppc 項を計算する方法としては、Read and Need の方法(RN 法) [3]と Kageshima and Shiraishi の方法(KS 法) [4]がある。本プログラムはこれらの方法を用いて ppc 項を計算できるようになっている。

11.2.1.6 Read and Needs 法

理論 [11,2]

RN 法はノルム保存型擬ポテンシャルに対する補正項を計算する方法である。電磁場が \mathbf{p} に及ぼす影響が小さいと仮定して、(26)式の $V(\mathbf{r}, \mathbf{p} + e\mathbf{A})$ を次のように展開する。

$$V(\mathbf{r}, \mathbf{p} + e\mathbf{A}) = V(\mathbf{r}, \mathbf{p}) + \frac{\partial V}{\partial \mathbf{p}} e\mathbf{A} \quad (28)$$

$\frac{\partial V}{\partial \mathbf{p}}$ を求めるため、 $V\phi$ を \mathbf{p} で微分する。 ϕ は任意の関数である。

$$\frac{\partial}{\partial \mathbf{p}} V\phi = \frac{\partial V}{\partial \mathbf{p}} \phi + V \frac{\partial \phi}{\partial \mathbf{p}} \quad (29)$$

ゆえに、

$$\frac{\partial V}{\partial \mathbf{p}} = \frac{\partial}{\partial \mathbf{p}} V + V \frac{\partial}{\partial \mathbf{p}} \quad (30)$$

ここで、 $i\hbar \frac{\partial}{\partial \mathbf{p}} = \mathbf{r}$ と定義する。定義された \mathbf{r} は交換関係 $[\mathbf{p}_\alpha, \mathbf{r}_\beta] = -i\hbar \delta_{\alpha\beta}$ を満たす。 α, β は座標インデックス

(x, y, z) である。この定義を用いて(30)式を書き直すと次式を得る。

$$\frac{\partial V}{\partial \mathbf{p}} = \frac{\partial}{\partial \mathbf{p}} V + V \frac{\partial}{\partial \mathbf{p}} = \frac{1}{i\hbar} [\mathbf{r}, V] = \frac{i}{\hbar} [V, \mathbf{r}] \quad (31)$$

(28)及び(31)式を用いて補正項を計算すると、以下の式を得る。

$$ppc = \frac{i}{\hbar} [V, \mathbf{r}] e\mathbf{A} \quad (32)$$

従って、摂動ハミルトニアンは、

$$H_{int} = (e/m)\mathbf{A} \cdot \mathbf{p} + \frac{i}{\hbar} [V, \mathbf{r}] e\mathbf{A} = (e/m)\mathbf{A} \cdot \left(\mathbf{p} + \frac{im}{\hbar} [V, \mathbf{r}] \right) \quad (33)$$

となる。対応する ϵ_2 の計算式は、(20)式において $\mathbf{p} \rightarrow \mathbf{p} + \frac{im}{\hbar} [V, \mathbf{r}]$ と置き換えることにより得られる。

$$\epsilon_2 = \frac{8\pi e^2 \hbar^2}{m^2 V} \sum_{\mathbf{k}_a, c, v} \frac{\left| \langle \Psi(\mathbf{k}_a) | \mathbf{u} \cdot \left(\mathbf{p} + \frac{im}{\hbar} [V, \mathbf{r}] \right) | \Psi(\mathbf{k}_a) \rangle \right|^2}{E_c(\mathbf{k}_a) - E_v(\mathbf{k}_a)} \delta(E_c(\mathbf{k}_a) - E_v(\mathbf{k}_a) - \hbar\omega) \quad (34)$$

\mathbf{r} 表示の計算式は、(23)式において、

$$\langle \Psi_c(\mathbf{k}_a) | \mathbf{r} | \Psi_v(\mathbf{k}_a) \rangle = \frac{1}{i\omega_{cv} m} \langle \Psi_c(\mathbf{k}_a) | \mathbf{p} | \Psi_v(\mathbf{k}_a) \rangle + \frac{1}{\hbar\omega_{cv}} \langle \Psi_c(\mathbf{k}_a) | [V, \mathbf{r}] | \Psi_v(\mathbf{k}_a) \rangle \quad (35)$$

とすることにより得られる。 ω_{cv} は次式で定義される量である。

$$\omega_{cv} = \frac{1}{\hbar} (E_c(\mathbf{k}_a) - E_v(\mathbf{k}_a)) \quad (36)$$

本プログラムでは、(23)及び(34)式を用いて電子状態を計算する。(35)式の右辺第1項をローカル項、第2項をノンローカル項あるいは ppc 項と呼ぶ。

計算法

ローカル項は、 Ψ_c 及び Ψ_v より直接計算できる。

$$\frac{1}{i\omega_{cv} m} \langle \Psi_c(\mathbf{k}_a) | \mathbf{p} | \Psi_v(\mathbf{k}_a) \rangle = \frac{1}{i\omega_{cv} m} \langle \Psi_c(\mathbf{k}_a) | \frac{\hbar}{i} \nabla | \Psi_v(\mathbf{k}_a) \rangle \quad (37)$$

$$\Psi_c(\mathbf{k}_a, \mathbf{r}) = \frac{1}{\sqrt{V_u}} \sum_{\mathbf{G}} \phi_{c, \mathbf{k}_a + \mathbf{G}} \exp(i(\mathbf{k}_a + \mathbf{G}) \cdot \mathbf{r}) \quad (38)$$

$$\Psi_v(\mathbf{k}_a, \mathbf{r}) = \frac{1}{\sqrt{V_u}} \sum_{\mathbf{G}} \phi_{v, \mathbf{k}_a + \mathbf{G}} \exp(i(\mathbf{k}_a + \mathbf{G}) \cdot \mathbf{r}) \quad (39)$$

ここで、 V_u は結晶ユニットセルの体積、 \mathbf{G} は平面波基底の \mathbf{G} ベクトル、 ϕ は展開係数である。(37)及び(38)式を(36)に代入し、平面波の直交条件を用いるとローカル項の計算式を得る ((40)式)。本プログラムは誘電体の電子バンド構造計算の結果得られる Ψ_c 及び Ψ_v よりローカル項を計算する。

$$\frac{1}{i\omega_{cv} m} \langle \Psi_c(\mathbf{k}_a) | \mathbf{p} | \Psi_v(\mathbf{k}_a) \rangle = \frac{\hbar}{i\omega_{cv} m} \sum_{\mathbf{G}} \phi_{c, \mathbf{k}_a + \mathbf{G}}^* \phi_{v, \mathbf{k}_a + \mathbf{G}} (\mathbf{k}_a + \mathbf{G}) \quad (40)$$

ノンローカル項の計算は、交換関係 $[V_{nl}, \mathbf{r}]$ を評価することにより行う。疑ポテンシャルのノンローカル部分は(40)式のようにあらわすことができる。

$$V_{nl} = \sum_{nml} |n, l\rangle D_{nm}^l \langle m, l| \quad (41)$$

$|n, l\rangle$ 及び $\langle m, l|$ は疑ポテンシャルのノンローカルプロジェクター、 D は係数、 l は原子のインデックスである。ノンローカル項は次式によりあらわされる。 Ψ_c 及び Ψ_v の波数ベクトル \mathbf{k}_a は省略してある。

$$\frac{1}{\hbar\omega_{cv}} \langle \Psi_c | [V, \mathbf{r}] | \Psi_v \rangle = \frac{1}{\hbar\omega_{cv}} \sum_{nll} \langle \Psi_c | n, l \rangle D_{nm}^l \langle l, l | \mathbf{r} | \Psi_v \rangle \quad (42)$$

$$- \frac{1}{\hbar\omega_{cv}} \sum_{nml} \langle \Psi_c | \mathbf{r} | n, l \rangle D_{nl}^l \langle l, l | \Psi_v \rangle \quad (43)$$

本プログラムは、Pickard and Payne の方法 [12] ((44)式) により $\langle n, l | \mathbf{r} | \Psi \rangle$ を計算し、ノンローカル項を計算する。 $\omega_{cv}, \langle l, l | \Psi \rangle$ は電子バンド構造計算により得られる値を用いる。

$$\langle n, l | \mathbf{r}_\alpha | \Psi \rangle \geq \frac{1}{2i|q|} [\langle n, l | e^{iq_\alpha \mathbf{r}} | \Psi \rangle - \langle n, l | e^{-iq_\alpha \mathbf{r}} | \Psi \rangle] \quad (44)$$

α はカーテシアンインデックス(= x, y, z)であり、 \mathbf{q}_α は次式で定義されるベクトルである。

$$\mathbf{q}_x = (q, 0, 0); \mathbf{q}_y = (0, q, 0); \mathbf{q}_z = (0, 0, q) \quad (45)$$

q はパラメータであり、微小な数である。

11.2.1.7 Kageshima and Siraishi 法

理論 [3]

KS 法における遷移モーメントは(46)式で与えられる。

$$\langle \varphi_c(\mathbf{k}_a) | \mathbf{r} | \varphi_v(\mathbf{k}_a) \rangle = \frac{1}{i\omega_{cv}m} \langle \varphi_c(\mathbf{k}_a) | \mathbf{p} | \varphi_v(\mathbf{k}_a) \rangle + \frac{1}{i\hbar\omega_{cv}} \sum_{nl} \langle \varphi_c | n, l \rangle \mathbf{p}_{nl}^l \langle l, l | \varphi_v \rangle \quad (46)$$

右辺の第一項はローカル項である。右辺の第二項が ppc 項である。 \mathbf{p}_{nl}^l は次式で定義される量である。

$$\mathbf{p}_{nl}^l = \langle \phi_n^l | \mathbf{p} | \phi_n^l \rangle - \langle \psi_l^l | \mathbf{p} | \psi_l^l \rangle \quad (47)$$

ϕ^l は全電子計算により得られる原子 l の軌道、 ψ^l は擬原子 l の軌道、 nl は軌道インデックスである。

計算法

本プログラムは、擬ポテンシャル計算プログラム CIAO の出力する \mathbf{p}_{nl}^l を読み込み(46)式の ppc 項を計算する。 $\omega_{cv}, \langle n, l | \varphi \rangle$ は電子バンド計算により得られた値を用いる。

11.2.1.8 電子系誘電率

電子系誘電率の計算式 ((23)式) は波数ベクトル \mathbf{k}_a に関する和を含む。本プログラムは \mathbf{k}_a に関する和を積分に置き換え、 ϵ_2 を求める ((48)式)。

$$\epsilon_2 = \frac{e^2}{\pi^2} \sum_{c,v} \int |\langle \Psi(\mathbf{k}_a) | \mathbf{u} \cdot \mathbf{r} | \Psi(\mathbf{k}_a) \rangle|^2 \delta(E_c(\mathbf{k}_a) - E_v(\mathbf{k}_a) - \hbar\omega) d\mathbf{k}_a \quad (48)$$

積分は、Linear Tetrahedron 法を用いて行う。Linear Tetrahedron の詳細については文献 [5]を参照されたい。 ϵ_2 をクラマース・クロニッヒ変換 ((24)式) することにより ϵ_1 を得る。クラマース・クロニッヒ変換は(24)式を台形公式を用いて数値積分することにより求める。

11.2.1.9 非線形光学感受率

材料の分極を P 、外部電場を F とした場合、一般に P は F の多項式であらわすことができる((49)式)。通常の誘電応答は、 F に関する 2 次以上の高次項を無視し、 P は F の 1 次関数であると近似して記述できる。しかし、レーザ光のように電場強度が強い電磁波を材料に照射した場合、高次項の P に関する寄与が無視できなくなり、高次項に起因する誘電・光学現象を顕著となる。高次項に起因する誘電・光学現象が非線形光学効果(nonlinear optical effect)である [13]。

$$P_i = \sum_j \chi_{ij}^{(1)} F_j + \sum_{jk} \chi_{ijk}^{(2)} F_j F_k + \sum_{jkl} \chi_{ijkl}^{(3)} F_j F_k F_l \quad (49)$$

ここで、 i, j, k, l は XYZ 座標のインデックス、 $\chi^{(1)}$ は線形感受率テンソル、 $\chi^{(2)}$ 及び $\chi^{(3)}$ は、それぞれ 3 階及び 4 階の非線形光学テンソルである。 $\chi^{(2)}$ に起因する現象を 2 次非線形光学効果、 $\chi^{(3)}$ に起因する現象を 3 次非線形光学効果と呼ぶ。材料の $\chi^{(2)}$ 及び $\chi^{(3)}$ は、誘電率同様、電子系及び格子系からの成分からなり、入射光の振動数が材料の格子振動数よりも低い場合には電子系と格子系が、光振動数が格子振動数よりも高い場合には電子系のみが実測値に寄与する。レーザ光の振動数は、格子振動数よりも高いため、非線形光学では電子系の $\chi^{(2)}$ 及び $\chi^{(3)}$ が興味の対象である。

UVSOR は、第 2 高調波発生 (Second Harmonic Generation (SHG)) 及び第 3 高調波発生 (Third Harmonic Generation (THG)) の感受率を計算する。SHG は、材料に入射された光子が 2 個結合し、周波数が倍の光が発生する 2 次非線形光学効果である。その感受率 $\chi^{(2)}(-2\omega; \omega, \omega)$ は複素数であり ((50)式)、電場を摂動ハミルトニアンとする 3 次の時間依存摂動法を用いて、固体の電子バンド構造より計算できる。

$$\chi^{(2)}(-2\omega; \omega, \omega) = \chi^{(2)'}(-2\omega; \omega, \omega) + i\chi^{(2)''}(-2\omega; \omega, \omega) \quad (50)$$

摂動計算では、価電子の伝導バンドへの仮想励起と伝導帯ホールと価電子バンドへの仮想励起を考える。電子及び正孔の仮想励起を図 11-2 に模式的に示す。

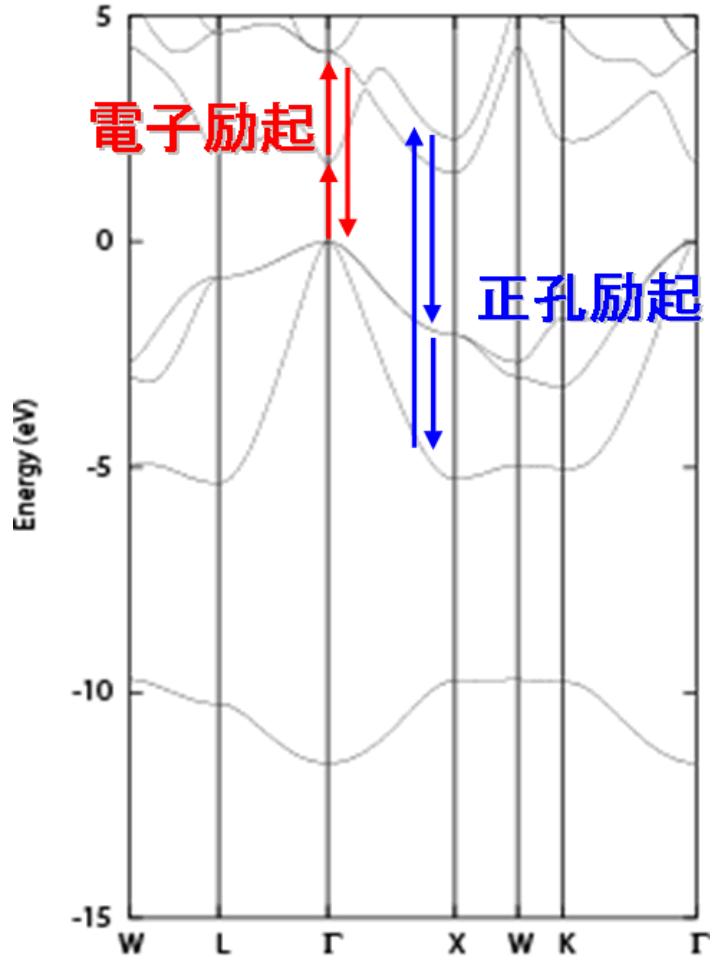


図 11-2 SHG 過程における電子と正孔の仮想励起

$\chi^{(2)'}(-2\omega; \omega, \omega)$ は電子励起による $\chi_{VE}^{(2)'}(-2\omega; \omega, \omega)$ と正孔励起による $\chi_{VH}^{(2)'}(-2\omega; \omega, \omega)$ の和で与えられ
((51)式)、 $\chi_{VE}^{(2)'}(-2\omega; \omega, \omega)$ と $\chi_{VH}^{(2)'}(-2\omega; \omega, \omega)$ はそれぞれ(52)及び(53)式により計算される [14]。

$$\chi^{(2)'}(-2\omega; \omega, \omega) = \chi_{VE}^{(2)'}(-2\omega; \omega, \omega) + \chi_{VH}^{(2)'}(-2\omega; \omega, \omega) \quad (51)$$

$$\chi_{VE}^{(2)'}(-2\omega; \omega, \omega) = -\frac{\pi}{2} \left| \frac{e\hbar}{m} \right| \sum_{i,j,l} \int_{BZ} \frac{d\mathbf{k}}{4\pi^3} \left(\frac{\Im[\mathbf{p}_{jl}^{cc}\{\mathbf{p}_{ij}^{cv}\mathbf{p}_{li}^{vc}\}]}{E_{li}^3(E_{li} + E_{ji})} \delta(E_{li} - \hbar\omega) \right. \\ \left. - \frac{\Im[\mathbf{p}_{ij}^{vc}\{\mathbf{p}_{jl}^{cc}\mathbf{p}_{li}^{cv}\}]}{E_{li}^3(2E_{li} - E_{ji})} \delta(E_{li} - \hbar\omega) + \frac{16\Im[\mathbf{p}_{ij}^{vc}\{\mathbf{p}_{jl}^{cc}\mathbf{p}_{li}^{cv}\}]}{E_{ji}^3(2E_{li} - E_{ji})} \delta(E_{ji} - 2\hbar\omega) \right) \quad (52)$$

$$\chi_{VH}^{(2)'}(-2\omega; \omega, \omega) = \frac{\pi}{2} \left| \frac{e\hbar}{m} \right| \sum_{i,j,l} \int_{BZ} \frac{d\mathbf{k}}{4\pi^3} \left(\frac{\Im[\mathbf{p}_{li}^{vv}\{\mathbf{p}_{ij}^{vc}\mathbf{p}_{jl}^{cv}\}]}{E_{jl}^3(E_{jl} + E_{ji})} \delta(E_{jl} - \hbar\omega) \right. \\ \left. - \frac{\Im[\mathbf{p}_{ij}^{vc}\{\mathbf{p}_{jl}^{cv}\mathbf{p}_{li}^{vv}\}]}{E_{jl}^3(2E_{jl} - E_{ji})} \delta(E_{jl} - \hbar\omega) + \frac{16\Im[\mathbf{p}_{ij}^{vc}\{\mathbf{p}_{jl}^{cv}\mathbf{p}_{li}^{vv}\}]}{E_{ji}^3(2E_{jl} - E_{ji})} \delta(E_{ji} - 2\hbar\omega) \right) \quad (53)$$

m は電子質量、 e は素電価、 c, v はそれぞれ伝導及び価電子バンドのインデックス、 \mathbf{p}_{ij} は座標表示の遷移モーメントより計算される運動量表示遷移モーメントの行列成分((54)式)、 E_{ab} はバンド間遷移エネルギー $E_{\mathbf{k}}^a - E_{\mathbf{k}}^b$ である。 \Im は虚部をとることを意味する。積分は、すべての \mathbf{k} 点について被積分関数の和をとることを意味する。誘電関数計算の場合と異なり、伝導バンド間および価電子バンド間の遷移が関与することに注意。

$$\mathbf{p}_{ij} = \langle \Psi_{\mathbf{k}}^i | \mathbf{p} | \Psi_{\mathbf{k}}^j \rangle = im(E_{\mathbf{k}}^c - E_{\mathbf{k}}^v) \langle \Psi_{\mathbf{k}}^i | \mathbf{r} | \Psi_{\mathbf{k}}^j \rangle \quad (54)$$

$\{\mathbf{p}_{ab}\mathbf{p}_{bc}\}$ は(55)式により計算されるテンソルである。 α 及び β はカーテシアン座標(x, y, z)のインデックスである。

$$\{\mathbf{p}_{ab}\mathbf{p}_{bc}\}_{\alpha\beta} = \frac{1}{2}(\mathbf{p}_{ab,\alpha}\mathbf{p}_{bc,\beta} + \mathbf{p}_{ab,\beta}\mathbf{p}_{bc,\alpha}) \quad (55)$$

(51)-(53)式により得られる $\chi^{(2) '}$ をクラマース・クロニツヒ変換((56)式)し、実部 $\chi^{(2)}$ を求める。

$$\chi^{(2) '}(-2\omega; \omega, \omega) = \frac{2}{\pi} P \int_0^\infty \frac{\Omega \chi^{(2)}(-2\Omega; \Omega, \Omega) '}{\Omega^2 - \omega^2} d\Omega \quad (56)$$

本プログラムでは、各 \mathbf{k} 点におけるバンド間の遷移モーメントを(54)式により求め、上記の計算法で感受率 $\chi^{(2)}(-2\omega; \omega, \omega)$ を求める。(52)及び(53)の積分は、Gaussian/parabolic smearing 法及び linear tetrahedron 法を用いて行う。Read and Needs 法(3.1.6)あるいは Kageshima and Shiraishi 法(3.1.7)を用いて遷移モーメントの補正を行った場合には、全電子計算と同じ結果が得られる。

(52)及び(53)式のデルタ関数の係数は分数であり、分母は 0 となりうる。デルタ関数の共鳴条件が成立し係数の分母が 0 となる場合、(52)及び(53)式の右辺は発散する。この発散は、2 重共鳴(double resonance)として知られる。2 重共鳴は、励起状態のダンピングファクターが 0 であると近似したために起きる現象である。本プログラムでは、係数の分母が一定の値(カットオフ値)よりも小さくなった場合、その項 $\chi^{(2)}$ のへの寄与を無視あるいはダンピングすることにより、2 重共鳴の問題を回避している。カットオフ値は、入力で与えるようになっている(4.2.13)。

THG は、材料に入射されたフォトンが 2 個結合し、周波数が 3 倍の光が発生する 3 次非線形光学効果である。THG 過程は、価電子帯の電子が伝導帯に散乱される過程(電子過程)、伝導帯の正孔が価電子帯に散乱される過程(正孔過程)、及び電子と正孔がそれぞれ同時に伝導帯及び価電子帯に散乱される過程(3 順位過程)からなる[21]。本プログラムは、光電場を摂動ハミルトニアンとする 4 次の時間依存摂動により感受率 $\chi^{(3)}(-3\omega; \omega, \omega, \omega)$ を計算する。

$$\text{Im}[\chi_{ve}^{(3)}(\omega)]_{\alpha\beta\gamma\delta} = -\frac{\pi}{3} \left(\frac{e\hbar}{m}\right)^4 \int_{BZ} \frac{d\mathbf{k}}{4\pi^3} \Sigma_{i,j,k,l} \text{Re}\{\mathbf{p}_{ij}^{vc}(\mathbf{p}_{jk}^{cc}, \mathbf{p}_{kl}^{cc}, \mathbf{p}_{li}^{cv})_{\beta\gamma\delta}\} f(E_{ji}, E_{ki}, E_{ji}, \hbar\omega) \quad (57)$$

$$\begin{aligned} \text{Im}[\chi_{vh}^{(3)}(\omega)]_{\alpha\beta\gamma\delta} = & -\frac{\pi}{3} \left(\frac{e\hbar}{m}\right)^4 \int_{BZ} \frac{d\mathbf{k}}{4\pi^3} \Sigma_{i,j,k,l} \left[\text{Re}\{\mathbf{p}_{ij}^{vc}(\mathbf{p}_{li}^{vv}, \mathbf{p}_{kl}^{vv}, \mathbf{p}_{jk}^{cv})_{\beta\gamma\delta}\} f(E_{ji}, E_{jl}, E_{jk}, \hbar\omega) \right. \\ & - \text{Re}\{\mathbf{p}_{ij}^{vc}(\mathbf{p}_{jk}^{cc}, \mathbf{p}_{li}^{vv}, \mathbf{p}_{kl}^{cv})_{\beta\gamma\delta}\} f(E_{ji}, E_{ki}, E_{kl}, \hbar\omega) \\ & \left. - \text{Re}\{\mathbf{p}_{ij}^{vc}(\mathbf{p}_{li}^{vv}, \mathbf{p}_{jk}^{cc}, \mathbf{p}_{kl}^{cv})_{\beta\gamma\delta}\} f(E_{ji}, E_{jl}, E_{kl}, \hbar\omega) \right] \end{aligned} \quad (58)$$

$$\begin{aligned} \text{Im}[\chi_{ts}^{(3)}(\omega)]_{\alpha\beta\gamma\delta} = & -\frac{\pi}{3} \left(\frac{e\hbar}{m}\right)^4 \int_{BZ} \frac{d\mathbf{k}}{4\pi^3} \Sigma_{i,j,k,l} \text{Re}\{\mathbf{p}_{ij}^{vc}(\mathbf{p}_{jk}^{cc}, \mathbf{p}_{kl}^{cc}, \mathbf{p}_{li}^{cv})_{\beta\gamma\delta}\} f(E_{ji}, E_{ki}, E_{ji}, \hbar\omega) \\ & \times \left[\frac{3^6}{E_{ji}^4(3kE_j - 2E_{ji})(3E_{li} - E_{jl})} \delta(E_1 - 3\hbar\omega) \right. \\ & + \frac{1}{E_{li}^4(E_{jk} + 3E_{li})} \\ & \left. + \frac{E_{ji} + E_{jk}}{(E_{ji} - 3E_{li})(E_{lk} + E_{li})} \right] \delta(E_{li} - \hbar\omega) \end{aligned} \quad (59)$$

ここで、

$$\begin{aligned} f(E_1, E_2, E_3, \hbar\omega) = & \frac{3^6}{E_1^4(3E_2 - 2E_1)(3E_3 - E_1)} \delta(E_1 - 3\hbar\omega) \\ & + \frac{1}{E_2^4(2E_3 - E_2)(2E_3 - 3E_2)(2E_1 + E_2)} \\ & + \frac{1}{E_3^4(E_2 - 2E_3)} \left(\frac{1}{E_1 - 3E_3} + \frac{2E_2}{(E_3 + E_1)(E_2 + 2E_3)} \right) \delta(E_3 - \hbar\omega) \end{aligned} \quad (60)$$

である。 α, β, γ , 及び δ は、カーテシアン座標 (x, y, z) のインデックスを意味する。 $(\dots)_{\beta\gamma\delta}$ は、遷移モーメント積のインデックス β, γ , 及び δ を対称化することを意味する。(57), (58), (59)式はそれぞれ、電子過程、正孔過程、及び 3 順位過程による $\chi^{(3)}$ の虚部を与える。 $\chi^{(3)}$ の虚部は、 $\text{Im}[\chi_{ve}^{(3)}]$, $\text{Im}[\chi_{vh}^{(3)}]$, 及び $\text{Im}[\chi_{ts}^{(3)}]$ の和で与えられる。 $\chi^{(3)}$ の虚部をクラマース・クロニツヒ変換することにより $\chi^{(3)}$ の実部を得る((61) 式)。

$$\text{Re}[\chi^{(3)}(-3\omega; \omega, \omega, \omega)] = \frac{2}{\pi} P \int_0^\infty \frac{\omega}{\omega'^2 - \omega^2} \text{Im}[\chi^{(3)}(-3\omega'; \omega', \omega', \omega'ight)] d\omega' \quad (61)$$

(57)-(59)式のブリルアン・ゾーン積分は Gaussian/parabolic smearing 法により計算する。2重共鳴条件の処理は、SHG 計算の場合同様、共鳴項を無視あるいはダンピングして行う。

11.2.1.10 有効質量

電子及び正孔の有効質量は、kp 摂動法を用いて次式により計算される [15]。

$$\hbar^2 \left(\frac{1}{m^*} \right)_{\alpha\beta} = \frac{\partial^2 E_{k\lambda}}{\partial k_\alpha \partial k_\beta} = \left[\delta_{\alpha\beta} + \frac{1}{m} \sum_{\lambda' \neq \lambda} \frac{(\mathbf{p}_{\lambda\lambda'}^k)_{pha} (\mathbf{p}_{\lambda'\lambda}^k)_\beta + (\mathbf{p}_{\lambda\lambda'}^k)_\beta (\mathbf{p}_{\lambda'\lambda}^k)_\alpha}{E_{k\lambda} - E_{k\lambda'}} \right] \quad (62)$$

\mathbf{k} 及び λ は有効質量計算を行う \mathbf{k} 点及びバンドのインデックス、 λ' は他のバンドのインデックス、 α 及び β は座標

インデックス、 $\mathbf{p}_{\lambda\lambda'}^k$ はバンド $\lambda\lambda'$ 間の遷移モーメント、 $E_{k\lambda}$ は計算を行う \mathbf{k} 点でのバンド λ のエネルギーである。

バンドがエネルギー的に縮退している場合、波動関数がバンド間で混ざり合うため、(62)式により縮退しているバンドの有効質量を計算すると、正しい値が得られない問題がある。この問題は、特に Γ 点での正孔の有効質量を計算する場合に顕著となる。本プログラムでは Γ 点での有効質量を、縮退が解けている Γ 点より僅かシフトした点で計算するようにして、この問題を回避している。シフト量の程度は入力データで指定する(4.2.9)。

11.2.2 格子系

11.2.2.1 概要

格子系誘電率計算プログラム Berry-Phonon は第一原理計算プログラム PHASE の拡張機能として実装されている。結晶の格子誘電率の計算には有効電荷と結晶の基準振動の振動数が必要である。Berry-Phonon では結晶の分極をベリー位相分極理論に基づき計算し、各原子のボルン有効電荷を求める。また、PHASE で計算されるヘルマン-フラインマン力を数値微分することにより力定数を計算し、これから動力学行列を構築して、その行列の固有値問題を解き、基準振動の振動数と固有ベクトルを求める。ボルン有効電荷と振動モードの固有ベクトルからモード有効電荷が求まる。基準振動の振動数とモード有効電荷から格子誘電率が計算される。この計算方法の詳細を以降の節で解説する。

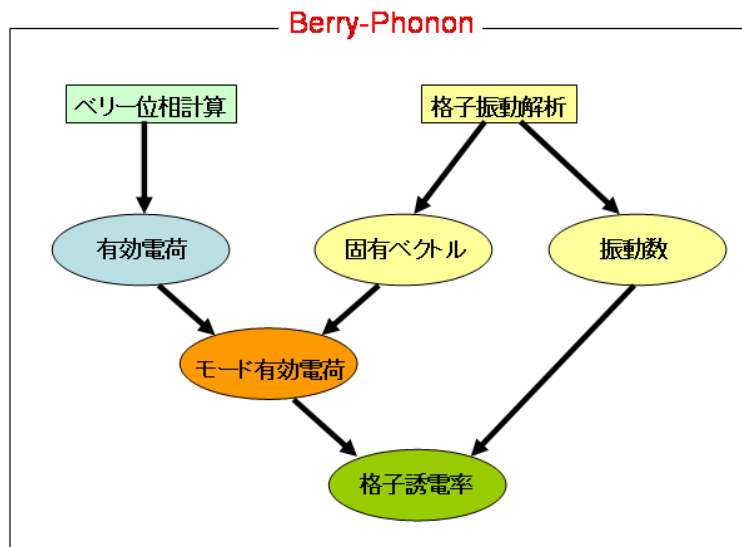


図 11-3 Berry-Phonon 構成図

11.2.2.2 格子誘電率

結晶の*i*番目の原子の変位ベクトルを \mathbf{u}_i とすれば、原子が平衡位置 からずれた際に発生する分極の変化は

$$\Delta \mathbf{P} = \frac{e}{V} \sum_i \mathbf{Z}_i^* \mathbf{u}_i \quad (63)$$

とあらわされる。ここで、 e は電気素量であり、 V は結晶の単位胞の体積である。 \mathbf{Z}_i^* はボルン有効電荷テンソルとよばれ、結晶単位胞中の各原子 に固有の物理量である。 原子の変位 \mathbf{u}_i を結晶の振動モードで分解することができる。

$$\sqrt{m_i} \mathbf{u}_{i\alpha} = \sum_{\lambda} \xi_{\lambda i \alpha} Q_{\lambda} \quad (64)$$

ここで、 $\xi_{\lambda i \alpha}$ は基準振動の固有ベクトルであり、 Q_{λ} は基準座標である。 m_i は*i*番目の原子の質量である。 分極の変化を基準座標で表現すれば

$$\Delta \mathbf{P} = \frac{e}{V} \sum_{\lambda} \tilde{\mathbf{Z}}_{\lambda} Q_{\lambda} \quad (65)$$

となる。ここで、振動モードの有効電荷を

$$\tilde{Z}_{\lambda \alpha} = \frac{1}{\sqrt{m_i}} \sum_{i\beta} Z_{i\alpha\beta}^* \xi_{\lambda i \beta} \quad (66)$$

と定義した。

振動数 ω の巨視的な電場を \mathbf{E} とすれば、モード有効電荷がゼロでない 基準振動の基準座標 Q_λ は電場に比例して振動する。

$$Q_\lambda = \frac{e\tilde{Z}_\lambda \cdot \mathbf{E}}{\omega_\lambda^2 - \omega^2} \quad (67)$$

格子誘電関数 $\epsilon^{\text{lat}}(\omega)$ は $4\pi\Delta\mathbf{P} = \epsilon^{\text{lat}}(\omega)\mathbf{E}$ で定義される。 この定義と(65)式と(67)式 から格子誘電関数は

$$\epsilon_{\alpha\beta}^{\text{lat}}(\omega) = \frac{4\pi e^2}{V} \sum_\lambda \frac{\tilde{Z}_{\lambda\alpha}\tilde{Z}_{\lambda\beta}}{\omega_\lambda^2 - \omega^2} \quad (68)$$

と表現できる。 THz 領域の誘電関数 $\epsilon_{\alpha\beta}(\omega)$ は格子誘電関数 $\epsilon^{\text{lat}}(\omega)$ に電子誘電率 $\epsilon_{\alpha\beta}^\infty$ を加えたものである。

$$\epsilon_{\alpha\beta}(\omega) = \epsilon_{\alpha\beta}^\infty + \frac{4\pi e^2}{V} \sum_\lambda \frac{\tilde{Z}_{\lambda\alpha}\tilde{Z}_{\lambda\beta}}{\omega_\lambda^2 - \omega^2} \quad (69)$$

11.2.2.3 ベリー一位相分極[16, 17, 18]

ボルン有効電荷を得るにはまず、結晶の分極を求める必要がある。 結晶の分極はイオンからの寄与 \mathbf{P}_{ion} と価電子からの寄与 \mathbf{P}_{el} とからなる。

$$\mathbf{P} = \mathbf{P}_{\text{ion}} + \mathbf{P}_{\text{el}} \quad (70)$$

イオンからの寄与は

$$\mathbf{P}_{\text{ion}} = \frac{e}{\Omega} \sum_l Z_l \mathbf{R}_l \quad (71)$$

である。 結晶の分極の変化は

$$\Delta\mathbf{P} = \mathbf{P}^{(1)} - \mathbf{P}^{(0)} \quad (72)$$

$$P_\alpha^{(\lambda)} = \frac{ifq_e}{8\pi^3} \sum_{n=1}^M \int_{\text{BZ}} d\mathbf{k} \langle u_{\mathbf{k}n}^{(\lambda)} | \partial / \partial k_\alpha | u_{\mathbf{k}n}^{(\lambda)} \rangle \quad (73)$$

とあらわせる。 ここで、 \mathbf{k}_\perp は \mathbf{G}_\parallel に垂直な面上のベクトルである。 \mathbf{k}_\perp を通り、 \mathbf{G}_\parallel に平行な長さ $|\mathbf{G}_\parallel|$ 線分を J 分割する点列 $\mathbf{k}_j = \mathbf{k}_\perp + j\mathbf{G}_\parallel/J$ ($j = 0, \dots, J-1$)を考える。このとき、 変数 $\phi_j^{(\lambda)}(\mathbf{k}_\perp)$ を以下のように定義する。

$$\phi_j^{(\lambda)}(\mathbf{k}_\perp) = \text{Im}\{\ln \Pi_{j=0}^{J-1} S^{(\lambda)}(\mathbf{k}_j, \mathbf{k}_{j+1})\} \quad (74)$$

$$S^{(\lambda)}(\mathbf{k}_j, \mathbf{k}_{j+1}) = \det (\langle u_{\mathbf{k}_j m}^{(\lambda)} | u_{\mathbf{k}_{j+1} n}^{(\lambda)} \rangle) \quad (75)$$

ここで、 $u_{\mathbf{k}_j n} = e^{-i\mathbf{G}_\parallel \cdot \mathbf{r}} u_{\mathbf{k}_0 n}$ である。これは $J \rightarrow \infty$ のとき \mathbf{k}_\perp を通る線分のベリー一位相となる。

$$\phi^{(\lambda)}(\mathbf{k}_\perp) = \lim_{J \rightarrow \infty} \phi_j^{(\lambda)}(\mathbf{k}_\perp) = -i \sum_{n=1}^M \int_0^{|\mathbf{G}_\parallel|} d\mathbf{k}_\parallel \langle u_{\mathbf{k}n}^{(\lambda)} | \frac{\partial}{\partial \mathbf{k}_\parallel} | u_{\mathbf{k}n}^{(\lambda)} \rangle \quad (76)$$

各 \mathbf{k} 点で独立に波動関数を計算したときにはベリー一位相以外の任意の位相ずれが許される。(77)式ではなく(76)式をもちいることにより、その位相ずれを打ち消すことができる。これより、分極の成分 $P_\parallel^{(\lambda)}$ は

$$P_\parallel^{(\lambda)} = -\frac{fq_e}{8\pi^3} \int_A d\mathbf{k}_\perp \phi^{(\lambda)}(\mathbf{k}_\perp) \quad (78)$$

とあらわせる。 \mathbf{b}_i 方向に沿って求められたベリー一位相を $\phi_i^{(\lambda)}$ とすれば分極は

$$\mathbf{P}^{(\lambda)} = -\frac{fq_e}{\Omega} \sum_i \frac{a_i}{2\pi} \phi_i^{(\lambda)} \quad (79)$$

とあらわせる。

ウルトラソフト擬ポテンシャルを用いた場合は電荷欠損補正を行う必要がある。(75)式に現れる積 $\langle u_{\mathbf{k}_j m}^{(\lambda)} | u_{\mathbf{k}_{j+1} n}^{(\lambda)} \rangle$ は $\psi_{\mathbf{k}n}^{(\lambda)}(\mathbf{r}) = e^{i\mathbf{k} \cdot \mathbf{r}} u_{\mathbf{k}n}^{(\lambda)}(\mathbf{r})$ を用いて、

$$M_{mn}(\mathbf{k}_j) = \langle \psi_{\mathbf{k}_j m}^{(\lambda)} | e^{-i\Delta\mathbf{k}\cdot\mathbf{r}} | \psi_{\mathbf{k}_j + \Delta\mathbf{k}, n}^{(\lambda)} \rangle \quad (80)$$

と表せる。ここで、 $\Delta\mathbf{k} = \mathbf{G}_{\parallel}/J$ である。ウルトラソフト擬ポテンシャルを用いた場合には電荷欠損を補うために(80)式の積の間に電荷密度演算子

$$K(\mathbf{r}) = |\mathbf{r}\rangle\langle\mathbf{r}| + \sum_l \sum_{ij} Q_{ij}^l(\mathbf{r}) |\beta_i^l\rangle\langle\beta_j^l| \quad (81)$$

を挿入しなければならない。ここで、 $l = \{R, \tau\}$ は原子位置を表すラベルである。

$$M_{mn}(\mathbf{k}_j) = \int d^3 r \langle \psi_{\mathbf{k}_j m}^{(\lambda)} | K(\mathbf{r}) e^{-i\Delta\mathbf{k}\cdot\mathbf{r}} | \psi_{\mathbf{k}_j + \Delta\mathbf{k}, n}^{(\lambda)} \rangle \quad (82)$$

(82)式から求まる(80)式に対する補正項を $M_{mn}^{\text{US}}(\mathbf{k})$ とする。

$$M_{mn}^{\text{US}}(\mathbf{k}) = \sum_l \sum_{ij} \int d^3 r Q_{ij}^l(\mathbf{r}) e^{-i\Delta\mathbf{k}\cdot\mathbf{r}} \langle \psi_{\mathbf{k} m}^{(\lambda)} | \beta_i^l \rangle \langle \beta_j^l | \psi_{\mathbf{k} + \Delta\mathbf{k}, n}^{(\lambda)} \rangle \quad (83)$$

$$= \sum_{\tau} \sum_{ij} \int d^3 r q_{ij}^{\tau}(\mathbf{r}) e^{-i\Delta\mathbf{k}\cdot\mathbf{r}} F_i^{\tau*}(m, \mathbf{k}) F_j^{\tau}(n, \mathbf{k} + \Delta\mathbf{k}) \quad (84)$$

最後の式は波動関数を平面波展開した場合($\psi_{\mathbf{k} n}^{(\lambda)}(\mathbf{r}) = \frac{1}{\Omega} \sum_{\mathbf{G}} c_{\mathbf{k} n, \mathbf{G}}^{(\lambda)} e^{i(\mathbf{k} + \mathbf{G})\cdot\mathbf{r}}$)の表現であり、

$$Q_{ij}^l(\mathbf{r}) = q_{ij}^{\tau}(\mathbf{r} - \boldsymbol{\tau} - \mathbf{R}) \quad (85)$$

$$\beta_i^l(\mathbf{r}) = \beta_i^{\tau}(\mathbf{r} - \boldsymbol{\tau} - \mathbf{R}) \quad (86)$$

$$F_i^{\tau}(n, \mathbf{k}) = \frac{1}{\sqrt{\Omega}} \sum_{\mathbf{G}} \int d^3 r \beta_i^{\tau}(\mathbf{r}) e^{i(\mathbf{k} + \mathbf{G})\cdot\mathbf{r}} e^{i\mathbf{G}\cdot\boldsymbol{\tau}} c_{\mathbf{k} n, \mathbf{G}}^{(\lambda)} \quad (87)$$

$$\langle \beta_i^l | \psi_{\mathbf{k} n}^{(\lambda)} \rangle = e^{-i\mathbf{k}\cdot(\mathbf{R} + \boldsymbol{\tau})} F_i^{\tau}(n, \mathbf{k}) \quad (88)$$

といった関係式を用いて格子和が取り除かれている。 $\psi_{\mathbf{k} + \mathbf{G}, n}^{(\lambda)} = \psi_{\mathbf{k} n}^{(\lambda)}$ なので、 $\langle \beta_i^l | \psi_{\mathbf{k} n}^{(\lambda)} \rangle = \langle \beta_i^l | \psi_{\mathbf{k} + \mathbf{G}, n}^{(\lambda)} \rangle$ が成り立つ。(88)式を適用すれば、 $F_i^{\tau}(n, \mathbf{k} + \mathbf{G})$ と $F_i^{\tau}(n, \mathbf{k})$ の間の関係が導ける。

$$F_i^{\tau}(n, \mathbf{k} + \mathbf{G}) = e^{-i\mathbf{G}\cdot\boldsymbol{\tau}} F_i^{\tau}(n, \mathbf{k}) \quad (89)$$

したがって、 $F_i^{\tau}(n, \mathbf{k}_j)$ は $F_i^{\tau}(n, \mathbf{k}_0)$ に位相因子 $e^{-i\mathbf{G}_{\parallel}\cdot\boldsymbol{\tau}}$ をかかけたものに等しい。

$$F_i^{\tau}(n, \mathbf{k}_j) = e^{-i\mathbf{G}_{\parallel}\cdot\boldsymbol{\tau}} F_i^{\tau}(n, \mathbf{k}_0) \quad (90)$$

11.2.2.4 ボルン有効電荷

結晶中のある原子のボルン有効電荷テンソル \mathbf{Z}^* はその原子の変位 \mathbf{u} によって生じた分極の変化 $\Delta\mathbf{P}$ とその変位との比例係数として定義される。

$$\Delta\mathbf{P} = -\frac{q_e}{\Omega}\mathbf{Z}^*\mathbf{u} \quad (91)$$

(70),(71),(79)を用いると、ボルン有効電荷は

$$Z_{\alpha\beta}^* = -\frac{\Omega}{q_e} \frac{\partial P_\alpha}{\partial u_\beta} \quad (92)$$

$$= Z_{ion}\delta_{\alpha\beta} + \sum_i \frac{f}{2\pi} a_{i\alpha} \cdot \frac{\partial \phi_i(u_\beta \hat{\beta})}{\partial u_\beta} \quad (93)$$

と表現できる。ここで、 \mathbf{a}_i は基本並進ベクトルであり、 $\phi_i(\mathbf{u})$ は逆格子ベクトル \mathbf{b}_i の方向に線積分を行った場合の原子変位 \mathbf{u} によるベリー一位相である。ベリー一位相の原子変位による微分は $\frac{\partial \phi_i(u_\beta \hat{\beta})}{\partial u_\beta}$ は差分近似で求める。原子変位 Δu_β によるベリー一位相の変化を $\Delta \phi_i$ とすると、

$$\frac{\partial \phi_i(u_\beta \hat{\beta})}{\partial u_\beta} = \frac{\Delta \phi_i}{\Delta u_\beta} \quad (94)$$

のように求める。

結晶中の原子は空間群の対称操作を行なうと等価な位置に移る。原子を動かさない対称操作の組から生成される点群がその原子の位置対称性をあらわす。有効電荷テンソルは位置対称性にしがいいゼロでない成分が決まる。位置対称性の対称操作を R_s とすれば、ボルン有効電荷 \mathbf{Z}^* は

$$\mathbf{Z}^* = \frac{1}{N} \sum_s R_s \mathbf{Z}^* R_s^{-1} \quad (95)$$

を満たさなければならない。ここで、 N は対称操作の数である。

等価原子のボルン有効電荷テンソルは等価原子に移す対称操作 $\{R|\mathbf{T}\}$ を作用させて求めることができる。

$$\mathbf{Z}_j^* = R \mathbf{Z}_i^* R^{-1} \quad (96)$$

$$\mathbf{r}_j = R \mathbf{r}_i + \mathbf{T} \quad (97)$$

ボルン有効電荷テンソルには零総和則があり、単位胞内の原子のボルン有効電荷 \mathbf{Z}_i^* の和をとるとゼロにならない。[19]

$$\sum_i \mathbf{Z}_i^* = 0 \quad (98)$$

\mathbf{k} 点数や平面波数に関する収束が不十分であると、ボルン有効電荷の零総和則が満たされなくなる。ボルン有効電荷の平均値

$$\bar{\mathbf{Z}}^* = \frac{1}{N_a} \sum_{i=1}^{N_a} \mathbf{Z}_i^* \quad (99)$$

を求め、ボルン有効電荷 \mathbf{Z}_i^* からボルン有効電荷の平均値 $\bar{\mathbf{Z}}^*$ を差し引くことで補正されたボルン有効電荷 $\mathbf{Z}_i^{*,new}$ を求めることができる。

$$\mathbf{Z}_i^{*,new} = \mathbf{Z}_i^* - \bar{\mathbf{Z}}^* \quad (100)$$

11.2.2.5 格子振動解析

格子振動解析の理論説明は 5.2.1 の振動解析の説明を参照されたい。

11.2.2.6 圧電応答

物質が歪むことにより、応力が発生する。歪みがわずかであれば、フックの法則が成り立ち、次式のように弾性定数 c_{ijkl} を用いて応力 σ_{ij} と歪み ϵ_{kl} が結びつけられる。

$$\sigma_{ij} = \sum_{kl} c_{ijkl} \epsilon_{kl} \quad (101)$$

ここで、 i, j, k, l はデカルト座標のインデックス x, y, z または $1, 2, 3$ である。誘電率が ϵ_{ij} の物質では電場 E_j と分極 P_i の間に次の関係が成り立つ。

$$P_i = \sum_j \frac{\epsilon_{ij} - \delta_{ij}}{4\pi} E_j \quad (102)$$

歪み ϵ_{kl} と電場 E_j が混在した場合は、式(101)には電場による項が加わり、式(102)には歪みによる項が加わる。圧電定数 $e_{i,kl}$ はひずみによる分極の変化率として定義される。

$$e_{i,kl} = \left(\frac{\partial P_i}{\partial \epsilon_{kl}} \right) \quad (103)$$

これより、(102)式は次のように修正される。

$$P_i = \sum_{kl} e_{i,kl} \epsilon_{kl} + \sum_j \frac{\epsilon_{ij} - \delta_{ij}}{4\pi} E_j \quad (104)$$

歪み ϵ_{kl} と応力 σ_{kl} のインデックス kl は短縮した表現 $1, 2, 3, 4, 5, 6$ で記述されることがある。その対応を下記に示す。

kl	11	22	33	23	31	12
短縮表現	1	2	3	4	5	6

以降の説明でも、適宜この表現を用いる。

次に電場による応力変化の表式を熱力学的考察により導く。分極の微小変化 dP_i によるエネルギー変化は $\sum_i E_i dP_i$ である。歪みの微小変化 $d\epsilon_{kl}$ によるエネルギー変化は $\sum_{kl} \sigma_{kl} d\epsilon_{kl}$ である。これらの和が自由エネルギー $F(T, \epsilon, P)$ の変化である。

$$dF = \sum_i E_i dP_i + \sum_{kl} \sigma_{kl} d\epsilon_{kl} \quad (105)$$

これに分極と電場との相互作用エネルギー $W = -\sum_i P_i E_i$ の変化 dW を加えてルジャンドル変換を行い、外場を含む自由エネルギー $F^*(T, \epsilon, E)$ の変化が求まる。

$$dF^* = dF + dW \quad (106)$$

$$= -\sum_i P_i dE_i + \sum_{kl} \sigma_{kl} d\epsilon_{kl} \quad (107)$$

これより、

$$\left(\frac{\partial F}{\partial E_i} \right) = -P_i \quad (108)$$

$$\left(\frac{\partial F}{\partial \epsilon_{kl}} \right) = \sigma_{kl} \quad (109)$$

であるから、次の関係が導かれる。

$$\left(\frac{\partial \sigma_{kl}}{\partial E_i} \right) = -\left(\frac{\partial P_i}{\partial \epsilon_{kl}} \right) = -e_{i,kl} \quad (110)$$

これより、(101)式は次のように修正される。

$$\sigma_{ij} = \sum_{kl} c_{ijkl} \epsilon_{kl} - \sum_k e_{k,ij} E_k \quad (112)$$

(104)式から電束密度は

$$D_i = 4\pi \sum_{kl} e_{i,kl} \epsilon_{kl} + \sum_j \epsilon_{ij} E_j \quad (113)$$

となる。この式と(112)式をあわせて圧電基本式という。

11.2.2.7 圧電応答のベリー位相理論[22]

(103)式によって定義される圧電定数はインプロパー圧電定数 $e_{i,kl}$ と呼ばれる。これとは違いプロパー圧電定数

$\tilde{e}_{i,kl}$ は、歪み速度 $\dot{\epsilon}_{kl} = \frac{d\epsilon_{kl}}{dt}$ による電流密度 J_i の変化率として与えられる。

$$\tilde{e}_{i,kl} = \frac{\partial J_i}{\partial \dot{\epsilon}_{kl}} \quad (114)$$

インプロパー圧電定数 $e_{i,kl}$ とプロパー圧電定数 $\tilde{e}_{i,kl}$ の間には次の関係が成り立つ。

$$\tilde{e}_{i,kl} = e_{i,kl} + \delta_{kl} P_i - \delta_{ik} P_l \quad (115)$$

ベリー位相分極理論によれば、電子からの結晶分極への寄与を波動関数をもとに計算することが可能である。ここでは、次式で与えられるように、ベリー位相を定義する。

$$\phi_\alpha^{el} = \frac{8\pi^3}{V} \sum_n \int_{BZ} d^3 k \langle u_{nk} | -i \mathbf{b}_\alpha \cdot \nabla_k | u_{nk} \rangle \quad (116)$$

α は逆格子ベクトル \mathbf{b}_α のインデックス 1,2,3 を表す。 u_{nk} は波動関数の周期的部分であり、 n はバンドインデックスであり、 \mathbf{k} は波数ベクトルである。 V は結晶の体積であり、積分記号の添え字 BZ はブリュアンゾーン内で積分することを表す。このとき、ベリー位相分極 \mathbf{P}_{el} は次式の様にあたえらる。

$$\mathbf{P}_{el} = -\frac{1}{2\pi V} \sum_\alpha \phi_\alpha^{el} \mathbf{a}_\alpha \quad (117)$$

ここで、 \mathbf{a}_α は基本並進ベクトルである。結晶分極のイオンからの寄与は次式のように与えられる。

$$\mathbf{P}_{ion} = -\frac{e}{V} \sum_\alpha Z_l \mathbf{R}_l \quad (118)$$

Z_l はイオンの電荷であり、 \mathbf{R}_l はイオンの位置ベクトルである。ここで、位相 $\phi_\alpha^{ion} = -\sum_l \mathbf{b}_\alpha \cdot \mathbf{R}_l$ を定義する。これとベリー位相 ϕ_α^{el} を加え合わせた位相を ϕ_α とする。これを用いて、結晶分極 \mathbf{P} は次のように書き表せる。

$$\mathbf{P} = -\frac{1}{2\pi V} \sum_\alpha \phi_\alpha \mathbf{a}_\alpha \quad (119)$$

この結晶分極 \mathbf{P} を歪みで微分したものはインプロパー圧電定数を与えることを示せる。ベリー位相分極は $\frac{e}{V} \mathbf{a}_\alpha$ の整数倍の不定性を持ち、そのためインプロパー圧電定数は一意に定まらない。しかし、プロパー圧電定数は一意に定まることが示せる。プロパー圧電定数の表式((120)式)は(119)式を(115)式に代入して求まる。

$$\tilde{e}_{i,kl} = -\frac{1}{2\pi V} \sum_\alpha \frac{\partial \phi_\alpha}{\partial \epsilon_{kl}} a_{\alpha,i} \quad (120)$$

11.2.2.8 圧電定数の計算方法

結晶の内部座標 u_j を固定して圧電定数を(120)式に従い計算したとき、その圧電定数をクランプドイオン近似プロパー圧電定数と呼ぶ。このとき位相 ϕ_α はベリー位相としてよい。クランプドイオン近似プロパー圧電定数は次式のように表せる。

$$\tilde{e}_{i,kl}^{(0)} = \left. \frac{\partial P_i}{\partial \epsilon_{kl}} \right|_{u_j=u_j^{(0)}} \quad (121)$$

ここで、 $u_j^{(0)}$ は歪みの無いとき内部座標を表す。結晶が歪んだときに内部座標が変化する効果を取り込むには、内部歪みパラメータ $\frac{\partial u_j}{\partial \epsilon_{kl}}$ とボルン有効電荷 $Z_l^* = \frac{V}{e} \frac{\partial \mathbf{P}}{\partial \mathbf{R}_l}$ に比例する量 $\frac{\partial P_i}{\partial u_j}$ を計算し、その積をクランプドイオン近似プロパー圧電定数に加えればよい。

$$\tilde{e}_{i,kl} = \tilde{e}_{i,kl}^{(0)} + \sum_j \frac{\partial P_i}{\partial u_j} \frac{\partial u_j}{\partial \epsilon_{kl}} \quad (122)$$

内部ひずみパラメータは力定数 $\Phi_{ma,j} = \frac{\partial F_{ma}}{\partial u_j}$ とひずみ-力結合定数 $C_{ma,kl} = \frac{\partial F_{ma}}{\partial \epsilon_{kl}}$ を用いて表せる。

$$\frac{\partial u_j}{\partial \epsilon_{kl}} = \sum_{ma} \frac{\partial u_j}{\partial F_{ma}} \frac{\partial F_{ma}}{\partial \epsilon_{kl}} \quad (123)$$

力定数は振動数と固有ベクトルを用いて表現できるので、(122)式の右辺第二項は次のように表すことができる。

$$\frac{e}{V} \sum_{\lambda} \frac{\tilde{Z}_{\lambda i} \tilde{C}_{\lambda, kl}}{\omega_{\lambda}^2} \quad (124)$$

ここで、振動モードのひずみ・力結合定数を

$$\tilde{C}_{\lambda, kl} = \frac{1}{\sqrt{m_i}} \sum_{i\beta} C_{i\beta, kl} \xi_{\lambda i\beta} \quad (125)$$

と定義した。

11.3 UVSOR-Epsilon

11.3.1 ファイルの設定

Epsilon は、PHASE/EKCAL と同様に、ファイル設定を `file_names.data` で行う。以下に Epsilon の `file_names.data` の例を示す。

```
&fnames
F_INP      = './nfinput.data'
F_POT(1)   = '../PP/atom_14_Si_lda_nc_bhs.gncpp2'
F_ENERG    = './nfenerg.data'
F_ZAJ      = './zaj.data'
F_CHGT     = '../scf/nfchgt.data'
F_CNTN     = './continue.data'
F_EPSOUT   = './eps.data1'
&end
```

`file_names.data` の設定法は PHASE/EKCAL の場合 [6] と同じであるが、以下の点に注意する。

1. 入力ファイルを `F_INP` に指定する。Epsilon の入力ファイルは EKCAL の入力ファイルに誘電率計算用の `epsilon` タグを追加したものである。`epsilon` タグの設定については 11.3.2.3 を参照。
2. PHASE 計算により得られた電子密度ファイルを `F_CHGT` に指定する。
3. 誘電関数の計算結果を出力するファイルを `F_EPSOUT` に指定する。

11.3.2 入力データの設定

入力ファイル (`F_INP`) ファイルの設定は、EKCAL の入力設定 [6] に Epsilon 用の設定を追加して行う。Epsilon の入力設定項目について説明する。

11.3.2.1 Control ブロック

control ブロックにおいて、`condition = 2` あるいは `condition=fixed_charge` とする。局所ポテンシャルが軌道ポテンシャルである TM 型擬ポテンシャルを用いる場合、`use_additional_projector = on` とする必要がある。Control ブロックの設定例を以下に示す。

```
Control{
  condition=fixed_charge
  cpumax = 1 day           ! {sec|min|hour|day}
  max_iteration = 60000
  use_additional_projector=on ! {on|off}
}
```

この例では `use_additional_projector=on` としている。ポテンシャルの種類が不明なときには `use_additional_projector` を on にする。

11.3.2.2 accuracy ブロック

accuracy ブロックで設定が必要な項目は、`num_bands` (バンド数)、`ksampling` (k 点セットの指定) 及び `ek_convergence` (バンド計算の収束条件) である。これらの項目の詳細は 2 章などに記述がある。

一般論としては、バンド数をできるだけ多く、k 点数をできるだけ多くすることが望ましいが、現実的には、1 電子あたりの振動子強度の総和 [7] が 0.7 を越えるように、バンド数及び k 点セットを設定すればよい。振動子強度の総和については、11.3.4.3 において説明する。

`ek_convergence` では、`delta_eigenvalue` 及び `succession` の設定を行う。`delta_eigenvalue` 及び `succession` の推奨値は以下の通り。

表 11.1 `ek_convergence` 設定の推奨値

材料系	delta_eigenvalue(Rydberg)	succession
絶縁体	1.e-4	3
半導体・金属	1.e-6	3

11.3.2.3 epsilon ブロック

epsilon ブロックにおいて、誘電関数の計算方法を指定する。以下に epsilon タグの例及びその設定方法を示す。

■ は必須の部分、■ の部分はオプション使用時に必須となる部分、他の部分は省略可能な部分である。

```
epsilon {
  sw_epsilon = on ! {on|off}
  crystal_type = single ! {single|poly}
  fermi_energy{
    read efermi = off ! {on|off}
    efermi = 0.000
  }
  photon{
    polar {ux=1.00, uy=0.00, uz=0.00}
    Poynting {px=0.00, py=0.00, pz=0.00}
    energy {low=0.000, high=2.000, step=0.002}
  }
  transition_moment{
    type = ks ! {l|rn|ks|mks}
    delq =0.001
    symmetry =on ! {on|off}
    band_i=1
    band_f=5
  }
  mass {
    sw_mass = on ! {on|off}
    direction{nx = 0.0, ny = 0.0, nz = 0.0}
    point = band_edge ! {band_edge|input}
    shift = 1.0d-4
    ik = 1
    ib = 5
  }
  BZ_integration {
    method = t ! {parabolic(p)|gaussian(g)|tetrahedron(t)}
    spin = both ! {both|major|minor}
  }
  band_gap_correction{
    scissor_operator=0.00d0
  }
  drude_term{
    drude= off ! {on|off|drude_only}
    effective_mass = 1.0d0
    damping_factor =
    conductivity =
    plasma_frequency =
  }
  nonlinear_optics {
    process = off ! {off|shg|thg}
    excitation = all ! {all|electron|hole|three_state}
    band = all ! {all|inter|intra}
    term = all ! {all|omega|2omega|3omega}
    double_resonance{
      method = damping !{omit|damping}
      cut_off = 10.0d-3 hartree
    }
  }
}
```

```

    }
  }
  ipriepsilon=1
}

```

11.3.2.4 sw_epsilon

機能：誘電関数の計算スイッチである。

オプション = on : 計算を行う

= off : 計算を行わない

効果 off とした場合、EKCAL として機能する (バンド構造の計算のみ行う)。

11.3.2.5 crystal_type

機能：結晶タイプの指定

オプション = single : 単結晶

= poly : 多結晶

poly を指定した場合、誘電率の異方性は平均化されているとみなし、単結晶の平均誘電率 $\epsilon = (\epsilon_{xx} + \epsilon_{yy} + \epsilon_{zz})/3$ を計算する。

11.3.2.6 fermi_energy

機能：フェルミレベルを指定する

パラメータ read_fermi

オプション = on : フェルミレベルを指定する

= off : フェルミレベルを指定しない (計算する)

efermi = : フェルミレベルの値を指定する (read_fermi = on 時)
(Hartree 単位で指定)

e.g. efermi = 0.124 (0.124 Hartree)

効果：read_fermi = on とすることにより、計算時間を短縮できる

注意：半導体および絶縁体の場合にのみ有効。

11.3.2.7 photon

機能：電磁波状態 (偏光状態およびエネルギーレンジ) の指定

polar : 直線偏光の分極ベクトルを指定する。

パラメータ ux = : 偏光ベクトルの x 成分 (任意単位)

uy = : 偏光ベクトルの y 成分 (任意単位)

uz = : 偏光ベクトルの z 成分 (任意単位)

e.g. polar{ux = 1.0, uy = 0.0, uz = 0.0} は、x 軸方向に直線偏光した電磁波を指定する。

Poynting : 非偏光のポインティングベクトルを指定する。

パラメータ px = : ポインティングベクトルの x 成分 (任意単位)

py = : ポインティングベクトルの y 成分 (任意単位)

pz = : ポインティングベクトルの z 成分 (任意単位)

e.g. Poynting {px = 1.0, py = 0.0, pz = 0.0} は、x 軸方向に進行する非偏光電磁波を指定する。

energy : 電磁波のエネルギーを指定する。

パラメータ high : エネルギー上限値 (Hartree 単位、デフォルト値 2.0)

step : エネルギーステップ (Hartree 単位、デフォルト値 0.002)

効果：指定された電磁波に対する誘電関数および光学スペクトル（屈折率、吸収係数、反射率）が計算される。

注意：i) 分極ベクトルのいずれかの成分が零でない場合、直線偏光が指定される。

ii) ポインティングベクトルのいずれかの成分が零でない場合、非偏光が指定される

iii) 分極ベクトルとポインティングベクトルを同時に指定することはできない。

iv) 分極ベクトルおよびポインティングベクトルの全ての成分が零である場合、誘電テンソル成分（ xx , yy , zz , xy , xz , yz ）が出力される。光学スペクトルは出力されない。

11.3.2.8 transition_moment

機能：遷移モーメント計算オプションの指定

パラメータ type：遷移モーメント補正方法の指定

オプション= 1：local 型遷移モーメント（補正なし）（デフォルト）

= rn：Read and Needs 型遷移モーメント補正[2,3]
（ノルム保存型擬ポテンシャル対応）

= ks：Kageshima-Shiraishi (KS) 型遷移モーメント補正[4]
（ノルム保存及びウルトラソフト擬ポテンシャル対応）

delq =：Read and Needs (RN) 型遷移モーメント補正のパラメータ[3]
（デフォルト値 0.001）

symmetry：遷移モーメントの対称化オプションの指定

オプション = on：対称化を行う

= off：対称化を行わない

band_i =：価電子バンドの指定

band_f =：伝導バンドの指定

効果：a) 遷移モーメントの補正方法を適切に指定することにより、全電子計算と同じ誘電関数を得ることができる。

b) band_i = a 及び band_f = b とした場合、a → b のバンド遷移に起因する誘電関数が計算され、誘電関数のバンド分割を行うことができる。

注意 i) KS 型遷移モーメント補正を行う場合、KS 補正因子を含む擬ポテンシャルファイル（gncpp2 形式）を用いる必要がある。補正因子を含むファイルは、CIAO のダイポールオプション（sw_with_dipole）を用いて作成できる。詳細は、CIAO のマニュアル[10]参照のこと。

ii) 局所ポテンシャルが軌道ポテンシャルである TM 型擬ポテンシャルを用いて KS 型遷移モーメント補正を行う場合、additional_projector を使用をする必要がある。

iii) RN 型補正は、ノルム保存型擬ポテンシャルに対してのみ有効である。補正項は Pickard and Payne の方法[12]により計算している。delq 値は補正項計算に必要な差分パラメータである。

iv) 結晶対称性を考慮した計算では、symmetry=on とすることがのぞましい。

v) 誘電関数のバンド分割を行わない場合、band_i 及び band_f は省略する。

11.3.2.9 mass

機能：電子あるいは正孔有効質量の計算

パラメータ sw_mass：計算スイッチ

オプション = on：計算を行う

= off：計算を行わない

direction：有効質量の方位を指定する

パラメータ nx、ny、nz：方位（デフォルト値 0.0）

vii) $\text{shift} \neq 0.0d0$ とした場合、誘電関数および非線形光学感受率に計算誤差が生じる。

オプション = on: ドルード項+バンド間遷移に起因する誘電関数を計算する。
 = off: バンド間遷移にのみ起因する誘電関数を計算する。
 (デフォルト)
 = drude_only: ドルード項にのみ起因する誘電関数を計算する。
 effective_mass = : 有効質量の指定 (電子質量単位)
 damping_factor = : ドルード damping factor の指定 (Hartree 単位)
 (デフォルト値=0.0036749 Hartree (=0.1eV))
 conductivity = : 電気伝導度の指定 (m/Ω)
 plasma_frequency = : プラズマ振動数の指定 (Hartree 単位)

効果: 金属のドルード項を考慮した誘電関数計算を行う

注意: i) 金属の場合にのみ有効である

ii) damping_factor、conductivity、及び plasma_frequency はいずれかひとつを指定できる。

iii) damping_factor パラメータを指定しない場合、デフォルト値 (0.0036749 Hartree) が適用される。

11.3.2.13 nonlinear_optics

機能: 非線形光学感受率の計算

パラメータ process: 非線形光学過程の指定

オプション = off: 感受率の計算を行わない (デフォルト)

= shg: 第2高調波発生の感受率計算を行う

= thg: 第3高調波発生の感受率計算を行う

excitation: 仮想励起プロセスの指定

オプション = all: 全ての励起プロセスの感受率を計算する (デフォルト)

= electron: 電子励起プロセスの感受率を計算する

= hole: 正孔励起プロセスの感受率を計算する

= three_state: 3 順位励起プロセスの感受率を計算する

band: バンド遷移の指定

オプション = all: 全バンド遷移の感受率を計算する (デフォルト)

= inter: バンド間遷移の感受率を計算する

= intra: バンド内遷移の感受率を計算する

term: 共鳴項の指定

オプション = all: 全共鳴項を考慮 (デフォルト)

= omega: 基本波に対する共鳴項を考慮する

= 2omega: 第2高調波に対する共鳴項を考慮する

= 3omega: 第3高調波に対する共鳴項を考慮する

double_resonance: 2重共鳴の扱いに関する指定

パラメータ method: 2重共鳴項の取り扱い方法を指定する

オプション = omit: 共鳴項を無視する (デフォルト)

= damping: 共鳴項をダンピングする

cut_off: 2重共鳴判定カットオフ (method = omit 指定時)

ダンピングファクター (method = damping 指定時)

デフォルト値: 10.0d-3 hartree

効果: 非線形光学感受率が計算される。

注意: i) ブリルアン・ゾーン積分法は、Bz_integration (2.10 参照) で指定する。THG 計算では、linear tetrahedron 法は使用できない。

ii) read_efermi = off が自動設定される (4.2.6 参照)

- iii) 感受率の定量的な計算には、scissors operator の指定 (7.2.11) が必要
- iv) 仮想励起及びバンド内遷移の意味は、文献 21 を参照

11.3.2.14 ipriepsilon

機能：プリントオプションの指定

オプション =0：簡略レベル
 =1：標準レベル（デフォルト）
 =2：詳細レベル
 =3：デバックレベル

効果：出力レベルを制御する。

注意：デバックレベルは出力データ量が非常に大きくなるので注意。

11.3.3 計算の実行

11.3.3.1 電荷密度の計算

Epsilon 計算は EKCAL と同様固定電荷の計算を行うので、電子密度ファイルが必要である。電子密度ファイルは、PHASE による電子密度の計算を行うと出力されるので、まずは PHASE による通常の計算を実行する。

Epsilon 計算を行うためには、電子密度ファイルを指定する必要がある。この指定は、file_names.data において F_CHGT 識別子を利用して行う。たとえば、電子密度ファイルが一階層上のディレクトリにおいて行われていた場合 file_names.data は次のように記述すればよい。

```
&fnames
F_POT(1) = ...
F_CHGT = '../nfchgt.data'
/
```

11.3.3.2 誘電関数の計算

以下のコマンドを実行することにより、Epsilon が実行される。

```
% mpirun -np 1 PATH_TO_PHASE0/bin/epsmain >& log &
```

並列計算を行う場合には、以下のコマンドを実行する。プロセッサの数を nproc で指定する。並列計算を行うにはあらかじめ MPI をインストールしておく必要がある。

```
% mpirun -np nproc PATH_TO_PHASE0/bin/epsmain >& log &
```

11.3.3.3 有効質量の計算

Epsilon 計算を行うディレクトリに電子密度ファイルをコピーし、同ディレクトリ中にある file_names.data において F_CNGT に指定する。入力ファイル nfinput.data における epsilon タグにおいて、有効質量計算に必要な入力を行う。

入力例 1：価電子帯端での電子有効質量テンソルの計算

```
mass{
    sw_mass = on      ! {on|off}
    direction {nx = 0.0, ny = 0.0, nz = 0.0}
    point = band_edge ! {band_edge|input}
    shift = 1.0d-4
}
```

入力例 2：伝導帯端での正孔有効質量((100)方向)の計算

```
mass{
  sw_mass = on      ! {on|off}
  direction {nx = 1.0, ny = 0.0, nz = 0.0}
  point = band_edge ! {band_edge|input}
  shift = 1.0d-4
}
```

以下のコマンドを実行することにより、Epsilon が実行される。

```
% mpirun -np 1 PATH_TO_PHASE0/bin/epsmain >& log &
```

11.3.3.4 非線形光学感受率の計算

Epsilon 計算を行うディレクトリに電子密度ファイルをコピーし、同ディレクトリ中にある file_names.data において F_CNGT に指定する。入力ファイル nfinput.data における nonlinear_optics において、必要な入力を行う。

以下のコマンドを実行することにより、Epsilon が実行される。

```
% mpirun -np 1 PATH_TO_PHASE0/bin/epsmain >& log &
```

11.3.4 計算結果の解析

計算結果の解析は、(1)遷移モーメントの計算状況、(2)電子状態、(3)振動子強度の総和則を確認して行う。これらの項目の確認は、誘電関数の計算結果が妥当であるかどうか確認する上で重要である。

11.3.4.1 遷移モーメントの計算状況

標準出力が output000 である場合、以下のコマンドを実行することにより、遷移モーメントの計算状況を確認できる。全ての k 点において遷移モーメントが計算されているかどうか確認することが必要である。

```
% grep transition output000
```

```
!* transition moment correction = Kageshima and Shiraishi method (1)
!* transition moment square matrix is symmetrized (2)
! PP transition moment correction data : it = 1 number of data read from PP file = 18
!* ----- transition moment of 1 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 2 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 3 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 4 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 5 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 6 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 7 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 8 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 9 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 10 -th k-point is calculated by UVSOR-Epsilon ----- (3)
!* ----- transition moment of 11 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 12 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 13 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 14 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 15 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 16 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 17 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 18 -th k-point is calculated by UVSOR-Epsilon -----
!* ----- transition moment of 19 -th k-point is calculated by UVSOR-Epsilon -----
!* transition moment of all k-points is calculated (4)
!* ----- weighted transition moment square of each k-point in irreducible Brillouin zone -----
integration of all possible band transitions (5)
!* tetrahedron integration of transition moment square over Brillouin zone (6)
```

各項目の意味は以下の通りである。

- (1) Kageshima-Shiraishi (KS) 型遷移モーメント補正を用いている
- (2) 遷移モーメントは結晶の対称性を反映している (transition_moment/symmetry = on である)
- (3) 各 k 点での電子遷移モーメント計算状況
- (4) 全ての k 点が収束している
- (5) 可能なすべての電子遷移を積分して誘電関数を計算する
- (6) 積分はリニアテトラヘドロン法を用いている

11.3.4.2 電子状態

各 k 点での遷移モーメントが計算された後、電子状態に関する状況が出力される

```

----- list of band type and occupation -----
ispin    band      type      occupation
  1         1      filled      1.00000
  1         2      filled      1.00000
  1         3      filled      1.00000
  1         4      filled      1.00000
  1         5      unfilled     0.00000
  1         6      unfilled     0.00000
  1         7      unfilled     0.00000
  1         8      unfilled     0.00000
  1         9      unfilled     0.00000      (7)
  1        10      unfilled     0.00000
  1        11      unfilled     0.00000
  1        12      unfilled     0.00000
  1        13      unfilled     0.00000
  1        14      unfilled     0.00000
  1        15      unfilled     0.00000
  1        16      unfilled     0.00000
  1        17      unfilled     0.00000
  1        18      unfilled     0.00000

----- list of band numbers for each spin -----

            filled    half-filled    unfilled    number of electrons
ispin = 1          4          0          14          4.00000 (8)

total number of electron in the system =    4.00000
The system is insulating or semiconducting (9)

```

(7) 各バンドのタイプ (filled: 被占バンド; half-filled: 金属バンド; unfilled: 空バンド) 及び電子占有数

(8) 各タイプのバンド数及び全電子数

(9) 対象系は、絶縁体あるいは半導体

11.3.4.3 振動子強度の総和則[20]

Thomas-Reiche-Kuhn's sum rule は、任意の体系において、全ての可能な励起に付随する振動子強度を足し合わせると、その値は1電子あたり1となることを主張する。実際のバンド計算では、有限のバンド数を用いて計算を行っているため、この sum rule が厳密に満たされることはない。しかし、sum rule は計算がどの程度現実の状況を反映しているかを表す指標となる。Epsilon は振動子強度の総和を計算する機能を有し、その値を出力する。

標準出力ファイルが output000 である場合、以下のコマンドを実行すると、振動子強度の総和値が画面に出力される。

% grep oscillator output000
!* sum of weighted oscillator strength of k-points in irreducible Brillouin zone = 0.91165


```
!* oscillator strength per electron = 0.91165
```

これはSi (num_bands=18; k 点セット=メッシュ法 (4x4x4)) の場合の計算値である。振動子強度の総和の値は約 0.9 であり、sum-rule が比較的良く満たされていることがわかる。 num_bands の値を増やすことにより振動子強度の総和は 1 に近づくが、経験的には、総和が 0.7 を越えていれば誘電関数は、ほぼ収束している場合が多い。

11.3.4.4 誘電関数の計算結果

計算により得られた計算結果は、file_names.data において EPS_OUTPUT に指定したファイルに出力される。以下の結果は、バルク Si 誘電関数の計算例である。

Dielectric Function			Optical Properties			
(1)	(2)	(3)	(4)	(5)	(6)	(7)
Photon Energy (eV)	Real Part	Imaginary Part	n	k	abs (in 10**8 m-1)	R
0.00000	13.90891	0.00000	3.72946	0.00000	0.00000	0.33307
0.05442	13.91137	0.00000	3.72979	0.00000	0.00000	0.33310
0.10885	13.91876	0.00000	3.73079	0.00000	0.00000	0.33320
0.16327	13.93110	0.00000	3.73244	0.00000	0.00000	0.33337
0.21769	13.94843	0.00000	3.73476	0.00000	0.00000	0.33361
0.27211	13.97078	0.00000	3.73775	0.00000	0.00000	0.33392

(以下略)

各カラムの意味は以下の通り。

- (1) 電磁波のエネルギー (2) 誘電関数 (実部) (3) 誘電関数 (虚部)
 (4) 屈折率 (実部) (5) 屈折率 (虚部) (6) 吸収係数 (7) 反射スペクトル

電磁波の分極ベクトル及びポインティングベクトルの成分を全て 0 として、誘電テンソルを出力した場合には、以下の出力が得られる。

Dielectric Tensor Component (Imaginary part is in parenthesis)						
Photon Energy (eV)	xx	yy	zz	xy	xz	yz
0.00000	13.90838	13.90838	13.90838	0.00000	0.00000	0.00000
	(0.00000)	(0.00000)	(0.00000)	(0.00000)	(0.00000)	(0.00000)
0.05442	13.91084	13.91084	13.91084	0.00000	0.00000	0.00000
	(0.00000)	(0.00000)	(0.00000)	(0.00000)	(0.00000)	(0.00000)
0.10885	13.91824	13.91824	13.91824	0.00000	0.00000	0.00000
	(0.00000)	(0.00000)	(0.00000)	(0.00000)	(0.00000)	(0.00000)
0.16327	13.93058	13.93058	13.93058	0.00000	0.00000	0.00000
	(0.00000)	(0.00000)	(0.00000)	(0.00000)	(0.00000)	(0.00000)
0.21769	13.94790	13.94790	13.94790	0.00000	0.00000	0.00000
	(0.00000)	(0.00000)	(0.00000)	(0.00000)	(0.00000)	(0.00000)

それぞれのカラムは、テンソル成分の誘電関数の実部及び虚部をあらわす。虚部は括弧内に示されている。

11.3.4.5 有効質量の計算結果

有効質量の計算結果は、標準出力ファイルに出力される。電子と正孔の有効質量は別々に計算することに注意する。

以下は、バルク Si の伝導帯端での電子有効質量テンソルを計算した例の出力である。

```
----- effective mass calculation -----
!* effective mass at valence band top: ik = 1
!* degeneracy = 3
!* warning : effective mass should be wrong because of the degeneracy.
!* set direction indices and k-point shift parameter in tag_mass.
!* ib = 2
!* ispin = 1
aa = -0.10765    bb = -0.36038    cc = -0.88554
      a          b          c
      0.70986    -0.07279    0.70057
      0.53349    -0.59385    -0.60227
      -0.45987    -0.80128    0.38272
```

```

!* ib =      3
!* ispin =    1
aa =  -0.09841    bb =  -0.47571    cc =  -1.07597          (1)
      a          b          c
      -0.55182    0.63785    0.53725
      0.52096    0.76670    -0.37519
      -0.65123    0.07285    -0.75538
!* ib =      4
!* ispin =    1
aa =  -0.10696    bb =  -0.29293    cc =  -2.32353
      a          b          c
      0.23175    0.97159    -0.04797
      -0.71433    0.13650    -0.68637
      -0.66032    0.19333    0.72567

!* effective mass at conduction band bottom: ik =    4
!* degeneracy =    1
!* ib =      5
!* ispin =    1
aa =   0.93658    bb =   0.18362    cc =   0.18362
      a          b          c
      1.00000    0.00000    0.00000          (2)
      0.00000    1.00000    -0.00235
      0.00000    0.00235    1.00000

```

(1) 価電子帯端 (Γ 点) での正孔有効質量テンソル。 Γ 点ではバンドが縮退しているため、計算値には問題がある (←の警告参照)

(2) 電子の有効質量テンソルの主値 (aa, bb, cc) とそれらの主軸(a, b, c)の方位。主軸方位はxyz座標表示

以下は、正孔有効質量 ((100)方向) 計算を行った例の出力である。

```

----- effective mass calculation -----
!* effective mass at valence band top: ik =    1
!* degeneracy =    3
!* ib =      2
!* ispin =    1
mass along ( 1.00000  0.00000  0.00000) direction =  -0.17130
!* ib =      3
!* ispin =    1
mass along ( 1.00000  0.00000  0.00000) direction =  -0.27190   (3)
!* ib =      4
!* ispin =    1
mass along ( 1.00000  0.00000  0.00000) direction =  -0.27190

!* effective mass at conduction band bottom: ik =    4
!* degeneracy =    1
!* ib =      5
!* ispin =    1
mass along ( 1.00000  0.00000  0.00000) direction =   0.93658   (4)

```

(3) (100)方向の正孔有効質量

(4) (100)方向の電子有効質量

11.3.4.6 SHG 非線形光学感受率の計算結果

計算結果は、標準出力ファイル及び `file_names.data` において、`F_NLO` に指定したファイルに出力される。標準出力には、静的な非線形感受率テンソル $\chi^{(2)}(0)$ が出力される。出力形式は以下の通り。以下は、Wurzite 型 AlN の計算結果例である。この例では、`scissors operator` を使用してバンドギャップが実測と同じになるように、ギャップを補正している。また、ブリルアンゾーン積分に `parabolic smearing` 法を用いている。このため、誘電関数の結果は、バンドギャップを補正しなかった場合あるいはブリルアンゾーン積分に `linear tetrahedron` 法を用いた場合の結果と異なる。

```
Static SHG Susceptibility Tensor (10-8 esu)
SHG prprocess = all type excitation (1)
SHG term = all terms (2)
xxx = 0.00000 xxy = 0.00000 xxz = -0.04514
xyy = 0.00000 xyz = 0.00000 xzz = 0.00000
yxx = 0.00000 yxy = 0.00000 yxz = 0.00000 (3)
yyy = 0.00000 yyz = -0.04514 yzz = 0.00000
zxx = 0.08732 zxy = 0.00000 zxz = 0.00000
zyy = 0.08732 zyz = 0.00000 zzz = -0.92412
```

- (1) 全てのSHG過程（電子及び正孔励起）を考慮
- (2) 全ての共鳴条件（基本波及び倍波に対する共鳴）を考慮
- (3) $\chi^{(2)}(0)$ テンソル（xxx等は $\chi_{xxx}^{(2)}$ を意味する）

`F_NLO` に指定したファイルには、波長依存の $\chi^{(2)}(-2\omega; \omega, \omega)$ テンソル各成分の実部、虚部及び絶対値がカラム状に出力される。以下は、Wurzite 型 AlN の計算結果例である。

```
SHG susceptibility Tensor (10d-8 esu)
xxx(1) (2) (3) (4) (5)
Photon Energy(eV) real part imaginary part abs
0.00000 0.00000 0.00000 0.00000
0.05442 0.00000 0.00000 0.00000
0.10885 0.00000 0.00000 0.00000
(中略)
zzz
Photon Energy(eV) real part imaginary part abs
0.00000 -0.92412 0.00000 0.92412
0.05442 -0.92432 0.00000 0.92432
0.10885 -0.92494 0.00000 0.92494
0.16327 -0.92597 0.00000 0.92597
0.21769 -0.92741 0.00000 0.92741
0.27211 -0.92926 0.00000 0.92926
```

- (1) テンソルのインデックス
- (2) 基本波のエネルギー
- (3) $\chi^{(2)}(-2\omega; \omega, \omega)$ の実部
- (4) $\chi^{(2)}(-2\omega; \omega, \omega)$ の虚部
- (5) $\chi^{(2)}$ の絶対値

11.3.4.7 THG 非線形光学感受率の計算結果

計算結果は、SHG 計算の場合同様、`file_names.data` において `F_NLO` に指定したファイルに出力される。出力形式は以下の通り。以下は、バルク Si の THG 感受率を計算結果例である。`scissor operator` を用いてバンドギャップの補正を行い、ブリルアン・ゾーン積分に `parabolic smearing` 法を用いている

```
Static THG Susceptibility Tensor (10-12 esu)
THG prprocess = all type excitation (1)
excitation = inter + intraband (2)
THG term = all terms (3)
xxxx = 59.19956 xxxxy = 0.00000 xxxz = 0.00000
xxyy = 24.15228 xxyz = 0.00000 xxzz = 24.15228
xyyy = 0.00000 xyzz = 0.00000 xyz = 0.00000
```

xzzz =	0.00000	yxxx =	0.00000	yxyx =	24.15228	(4)
yxxz =	0.00000	yxyy =	0.00000	yxyz =	0.00000	
yxzz =	0.00000	yyyy =	59.19956	yyyz =	0.00000	
yyzz =	24.15228	yzzz =	0.00000	zxxx =	0.00000	
zxxxy =	0.00000	zxxxz =	24.15228	zxyy =	0.00000	
zxyz =	0.00000	zxzz =	0.00000	zyyy =	0.00000	
zyyz =	24.15228	zyzz =	0.00000	zzzz =	59.19956	

- (1) すべての仮想励起過程（電子、正孔、3準位）を考慮
- (2) すべてのバンド遷移（バンド間+バンド内遷移）を考慮
- (3) すべての共鳴条件（基本波及び倍波に対する共鳴）を考慮
- (4) $\chi^{(3)}(0)$ のテンソル（xxxxなどは $\chi_{xxxx}^{(3)}$ を意味する）

F_NLO に指定したファイルには、波長依存の $\chi^{(3)}(-3\omega; \omega, \omega, \omega)$ テンソル各成分の実部、虚部及び絶対値がカラム状に出力される。以下は、Si の $\chi^{(3)}$ 計算結果である。

THG susceptibility Tensor (10d-12 esu)				
Xxxx(1)	(2)	(3)	(4)	(5)
Photon Energy(eV)	real part	imaginary part	abs	
0.00000	59.19956	0.00000	59.19956	
0.05442	59.33822	0.00000	59.33822	
0.10885	59.75834	0.00000	59.75834	
0.16327	60.47272	0.00000	60.47272	
0.21769	61.50384	0.00000	61.50384	

- (1) テンソルのインデックス (2) 基本波のエネルギー (3) $\chi^{(3)}(-3\omega; \omega, \omega, \omega)$ 実部 (4) $\chi^{(3)}(-3\omega; \omega, \omega, \omega)$ 虚部 (5) $\chi^{(3)}$ の絶対値

11.3.5 計算例：Si2-電子誘電関数計算

11.3.5.1 電子密度の計算

インストールが完了したら、テスト計算を兼ねて、シリコン結晶の誘電関数及び光学スペクトルを計算してみましょう。入力は `samples/uvSOR/electron/Si` にあります。`samples/uvSOR/electron/Si` の下位には、`scf`、`eps` 及び `PP` という名称のディレクトリがあります。`scf` は `phase` による電荷密度計算用のディレクトリ、`eps` は `UVSOR-Epsilon` による誘電率計算用のディレクトリ、`PP` は `Si` 原子の擬ポテンシャルファイルを格納するディレクトリです。

最初に、`Si` 結晶の電子密度を `PHASE` で計算します。`scf` に移動してください。`scf` には以下のファイルが含まれています。

```
file_names.data
nfinput.data
```

`file_names.data` は `PHASE` の入出力ファイルを指定するファイルです。この入力例では、電荷密度を `/nfchgt.data` に出力する設定になっています。

```
&fnames
F_INP      = './nfinput.data'
F_POT(1)   = '../PP/atom_14_Si_lda_nc_bhs.gncpp2'
F_CHGT     = './nfchgt.data' ← 電荷密度ファイルの設定
&end
```

`nfinput.data` は `PHASE` により `Si` 結晶の電荷密度を計算するためのファイルです。計算条件は以下のように設定されています。

交換相関ポテンシャル：LDAPW91

バンド数：8

k 点セット：メッシュ法(4x4x4)

SCF 収束条件：`scf_convergence = 10-12 Hartree`; `succession = 3`

使用する擬ポテンシャルは、`PP` に格納されている `atom_14_Si_lda_nc_bhs.gncpp2` です。擬ポテンシャルの形式は以下の通りです。

交換相関ポテンシャル：LDAPW91

局所ポテンシャル：BHS 形式

以下のコマンドを入力して `PHASE` の計算を行います。

```
% mpirun -np 1 PATH_TO_PHASE0/bin/phase >& log
```

11.3.5.2 誘電関数の計算

電子密度の計算が終了したら、誘電関数の計算を行います。`eps` ディレクトリに移動してください。このディレクトリには以下のファイルが格納されています。

```
file_names.data
nfinput.data
```

`file_names.data` は `UVSOR` の入出力ファイル設定を行うファイル、`nfinput.data` は `UVSOR` の入力ファイルです。`file_names.data` は以下のように設定されています。

```
&fnames
F_INP      = './nfinput.data'           ← (1) 入力データファイルの設定
F_POT(1)   = '../PP/atom_14_Si_lda_nc_bhs.gncpp2' ← (2) 擬ポテンシャルファイルの設定
F_CHGT     = './scf/nfchgt.data'       ← (3) 電子密度ファイルの設定
F_EPSOUT   = './eps.data'             ← (4) 誘電関数出力ファイルの設定
&end
```

- (1) 入力ファイルは PHASE/EKCAL と同じ形式です。入力ファイル例は次節で説明します。
- (2) 擬ポテンシャルファイルは、電子密度計算の用いたものと同じです。
- (3) scf ディレクトリで PHASE を実行して得られた電子密度ファイルを指定します。
- (4) 誘電関数用の出力ファイルです。名称は任意です。

入力ファイルの設定

nfinput.data は、以下内容となっています。

```
Control{
  condition = 2 (1) ! {0|1|2|3}|{initial|continuation|fixed_charge|fix
ed_charge_continuation}
  cpumax = 1 day ! {sec|min|hour|day}
  max_iteration = 6000
  use_additional_projector = off
  nfstopcheck = 1
}

accuracy{
  cke_wavefunctions = 25.0 rydberg ! cke_wf
  cke_chargedensity = 100.0 rydberg ! cke_cd
  num_bands = 18 (2)
  ksampling{
    method = mesh ! {mesh|file|directin|gamma|monk}
    mesh{ nx= 4, ny = 4, nz = 4 }
  }
  smearing{
    method = tetrahedral ! {parabolic|tetrahedral}
    width = 0.001 hartree
  }
  xctype = ldapw91 ! ldapw91
  scf_convergence{
    delta_total_energy = 1.e-12 hartree
    succession = 3 !default value = 3
  }
  force_convergence{
    max_force = 0.1e-3
  }
  ek_convergence{
    num_extra_bands = 0
    num_max_iteration = 300
    sw_eval_eig_diff = on (3)
    delta_eigenvalue = 1.e-6 rydberg (4)
    succession = 3 (5)
  }
  initial_wavefunctions = matrix_diagon !{random_numbers|matrix_diagon}
  matrix_diagon{
    cke_wf = 20.00 rydberg ! cke_wf
  }
  initial_charge_density = file !{Gauss|Very_broad|pseudopotentialfile}
}

(中略)

epsilon {
  sw_epsilon = on a
  crystal_type = single ! {single|poly} b
  fermi_energy{
```

```

        read_efermi = off  c
        efermi = 0.0000
    }
    photon{
        polar      {ux=1.00, uy=0.00, uz=0.00 } d
        Poynting {px=0.00, py=0.00, pz=0.00}
        energy      {low=0.000, high=2.000, step=0.002} e
    }
    transition_moment{
        type = ks ! {l|rn|ks}  f
        delq = 0.001
        symmetry = on  g
    }
    BZ_integration {
        method = t !{parabolic(p)|gaussian(g)|tetrahedron(t)} h
    }
    band_gap_correction{
        scissor_operator=0.0d0 i
    }
    drude_term {
        drude = off  j
    }
    ipriepsilon = 1  k
}

```

(以下略)

(1) control タグにおいて condition=2 とします。

(2) num_bands を設定します。num_bands は通常の SCF 計算の場合よりも大きな値として、1 電子あたりの振動子強度が 0.7 以上になるようにします。詳細は次節で説明します。

(3) sw_eval_eig_diff=on とします。

(4) delta_eigenvalue 値を設定します。delta_eigenvalue の推奨値は、半導体・金属の場合は 10^{-6} Rydberg 程度、絶縁体の場合 10^{-4} Rydberg 程度です。

(5) succession=3 とします。

(6) epsilon タグを設定。epsilon タグで誘電関数の計算方法を指定します。この入力例における epsilon タグの a, b, c, d, e, f, g, h, i, k の意味は以下の通りです。

a 誘電関数の計算を行う

sw_epsilon=off とすると誘電関数の計算を行わないので注意してください。

b 単結晶として計算する

c フェルミレベルを計算する。

d 入射電磁波は直線偏光しており、偏光ベクトルは(1.0, 0.0, 0.0)(x 軸)の方向を向いている。

e 入射電磁波のエネルギーレンジは 0~2.0 Hartree であり、エネルギーステップは 0.002 Hartree である。

f 遷移モーメント補正を行い、全電子計算と同じ結果が得られようにする。補正法は Kageshima-Shiraishi(KS)法を用いる。

g リニアテトラヘドロン法を用いて誘電率の虚部を求める

h scissors operator 値を 0 とする(バンドギャップの補正を行わない)。

i 標準出力とする。

(注意)Read and Needs(RN)法は、ノルム保存型擬ポテンシャルの場合にのみ有効です。KS 法はノルム保存及びウルトラソフト擬ポテンシャルに有効ですが、CIAO により作成された KS 補正因子(Dipole 補正因子)を含む擬ポテンシャルファイルを使用することが必要です。詳細は、UVSOR-Epsilon 及び CIAO のマニュアルを参照ください。なお、この計算に用いる擬ポテンシャルは補正因子を含んでいます。

11.3.5.3 誘電関数の計算 1

eps ディレクトリで以下のコマンドを実行してみてください。UVSOR-Epsilon が実行されます。

```
% mpirun -np 1 PATH_TO_PHASE0/bin/epsmain >& log &
```

計算が終わったら、電子状態の収束を確認します。収束の確認は、以下のコマンドで行うことができます。

```
% grep converged output000
```

このコマンドを実行して、**!* all k-points are converged** と画面に表示されれば、電子状態は収束しています。
eps.data を見てみましょう。以下のような出力が得られているはずです。

Dielectric Function			Optical Properties			
(1)	(2)	(3)	(4)	(5)	(6)	(7)
Photon Energy (eV)	Real Part	Imaginary Part	n	k	abs (in 10**8 m-1)	R
0.00000	13.90891	0.00000	3.72946	0.00000	0.00000	0.33307
0.05442	13.91137	0.00000	3.72979	0.00000	0.00000	0.33310
0.10885	13.91876	0.00000	3.73079	0.00000	0.00000	0.33320
0.16327	13.93110	0.00000	3.73244	0.00000	0.00000	0.33337
0.21769	13.94843	0.00000	3.73476	0.00000	0.00000	0.33361
0.27211	13.97078	0.00000	3.73775	0.00000	0.00000	0.33392
(以下略)						

各カラムの意味は以下の通りです。

(1) 電磁波のエネルギー (2) 誘電関数(実部) (3) 誘電関数(虚部)

(4) 屈折率(実部) (5) 屈折率(虚部) (6) 吸収係数 (7) 反射率

電磁波のエネルギー=0 における誘電率が静的誘電率です。計算により得られた静的誘電率は 13.90 で、実測(11.7)よりも大きな値となります。これは、LDA 法が Si のバンドギャップを過少評価することに起因しています。

11.3.5.4 バンド数の設定法

計算の信頼性は、計算に用いるバンド数に大きく依存します。信頼性を確認するには、1 電子あたりの振動子強度の総和をしらべます。振動子強度の総和は、以下のコマンドを使用します。

```
% grep oscillator output000
```

このコマンドを実行すると

```
!* oscillator strength per electron = 0.91165
```

という表示が画面に現われます。

この表示は、この計算における 1 電子あたりの振動子強度の総和が約 0.91 であることを示しています。振動子強度の総和則(トーマス・ライヒ・クーンの総和則)は、1 電子あたりの振動子強度の総和が 1 になることを主張します。バンド数(num_bands パラメータ)を増やすにつれ、oscillator strength per electron の値は 1 に近づいていきますが、実際の計算では、この値が 0.7 を越えるように num_bands を設定すれば、誘電関数はほぼ収束していることが経験的にわかっています。この例では、1 電子あたりの振動子強度の総和が 0.7 以上となっており、バンド数は十分であることがわかります。

11.3.5.5 誘電関数の計算 2

次に、Read and Needs(RN)方式で、遷移モーメント補正を行い、Si の誘電率を計算してみましょう。epsilon タグの f の部分を type = rn として、同様に計算を行ってみます。計算を行うと eps.data が上書きされるので、コピーをとっておきます。

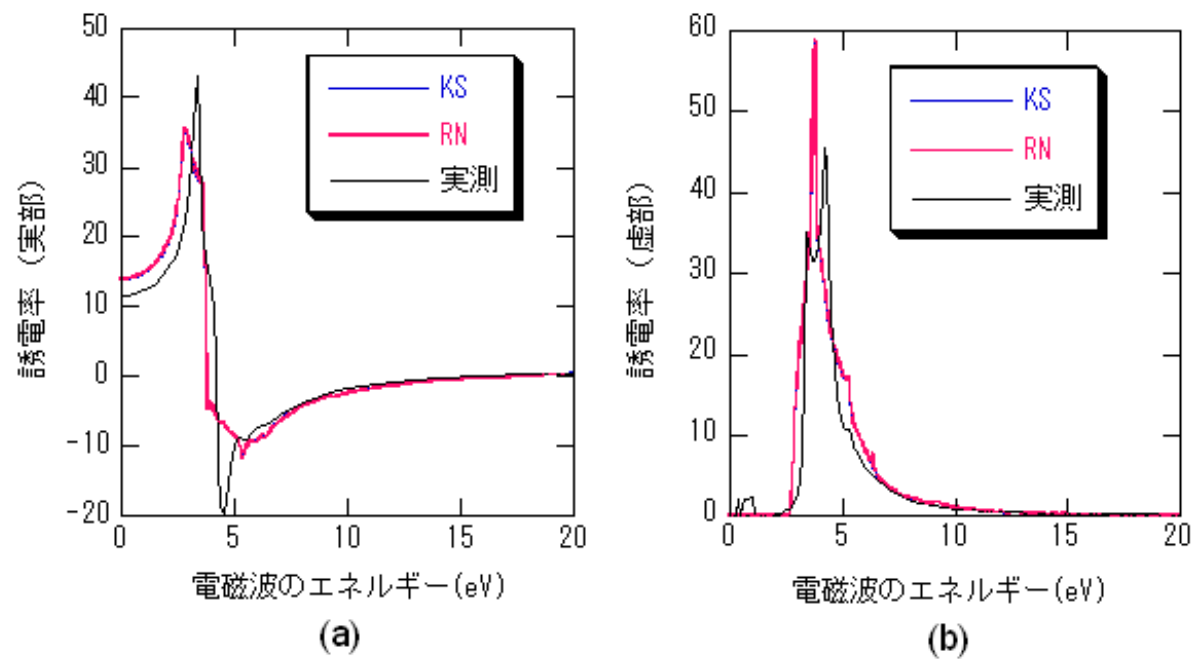
```
% cp eps.data eps.data-ks
```

同様に計算を行うと、以下のような結果が eps.data に出力されます。

Dielectric Function			Optical Properties			
Photon Energy (eV)	Real Part	Imaginary Part	n	k	abs (in 10**8 m-1)	R
0.00000	13.97263	0.00000	3.73800	0.00000	0.00000	0.33395
0.05442	13.97510	0.00000	3.73833	0.00000	0.00000	0.33398

0.10885	13.98253	0.00000	3.73932	0.00000	0.00000	0.33408
0.16327	13.99492	0.00000	3.74098	0.00000	0.00000	0.33425
0.21769	14.01231	0.00000	3.74330	0.00000	0.00000	0.33449
0.27211	14.03476	0.00000	3.74630	0.00000	0.00000	0.33480
(以下略)						

静的誘電率は 13.97 で、KS 法による計算値に極めて近い値となっています。3.2.3 で得られた KS 補正による誘電関数と、ここで得られた誘電関数を比較してプロットすると、以下ようになります。なお図 2 には、比較のため、光学測定により実測された誘電関数も示してあります。



Si 結晶の誘電関数：(a)実部 (b)虚部。KS 補正及びRN 補正をして計算した誘電関数を示す。実測の誘電関数（CRC Handbook of Chemistry 79-th Ed,CRC Press, New York 1998）を比較のため示す。

11.4 UVSOR-Berry-Phonon

11.4.1 入出力の説明

11.4.1.1 入力出力ファイル

PHASE と同様に入力および出力ファイルは `file_names.data` に記述して指定する。たとえば、以下のように記述する。

```
&fname  
F_INP      = './nfinput.data'  
F_POT(1)   = './potential.1'  
F_POT(2)   = './potential.2'  
F_CHGT     = './nfchgt.data'  
F_BERRY    = './berry.data'  
F_EFFCHG   = './effchg.data'  
F_FORCE    = './force.data'  
F_MODE     = './mode.data'  
F_EPSILON  = './epsilon.data'  
&end
```

F_INP は PHASE の入力ファイルであり、ベリー位相計算や振動解析計算の入力も このファイルに記述する。標準の PHASE には無いファイルについてのみ説明をする。F_BERRY はベリー位相計算の出力である。F_EFFCHG は通常は使用されないが、有効電荷を入力するファイルである。F_FORCE は振動解析に必要とされるファイルが出力されるファイルである。F_MODE は振動解析の結果およびモード有効電荷と誘電率が出力されるファイルである。F_EPSILON は誘電関数が出力されるファイルである。各入力および出力ファイルの説明を表 2 にまとめた。

表 2. ファイルポインタの説明

ファイルポインタ名	既定値	説明
F_INP	nfinp.data	入力ファイル。結晶構造、計算精度、計算の制御などの情報が記述されている。
F_BERRY	berry.data	ベリー位相の計算値が出力されるファイル。
F_EFFCHG	effchg.data	有効電荷を記述するファイル。
F_FORCE	force.data	振動解析に必要とされる力のデータが記述されるファイル。
F_MODE	mode.data	振動解析の結果およびモード有効電荷と誘電率が出力されるファイル。
F_EPSILON	epsilon.data	誘電関数が出力されるファイル。
F_STRFRC	strfrc.data	入力ファイル。内部座標を固定して結晶を歪ませた時に原子に作用する力を記述するファイル。

11.4.1.2 入力ファイル"F_INP"

"F_INP"で指定されるファイルには、結晶構造、計算精度、計算の制御などを記述する。格子誘電率計算は、Berry_phase ブロックと Phonon ブロックで主に制御される。Berry_phase ブロックは固有状態を計算するプログラム EKCAL でのみ有効で、ベリー位相計算の制御を行う。Phonon ブロックは PHASE でのみ有効で、格子振動解析の制御を行う。

Berry_phase ブロックの形式は次のようになっている。

```
Berry_phase{  
  sw_berry_phase = <ON_OFF>  
  g_index = <G_INDEX>  
  mesh{ n1 = <MESH_N1>, n2 = <MESH_N2>, J = <MESH_J> }  
}
```

表 3. BerryPhase 計算に関する変数の説明

変数名またはタグ名	既定値	説明
sw_berry_phase	OFF	ベリー位相計算を行うかどうかのスイッチ。
g_index	1	逆格子ベクトル \mathbf{g}_i ($i = 1, 2, 3$)の指数 i 。
mesh		\mathbf{k} 空間のメッシュを指定するブロックタグ。
n1,n2	4	選択した逆格子ベクトル \mathbf{g}_i に垂直な面内の Monkhorst-Pack メッシュの指数 $n1 \times n2$
J	20	選択した逆格子ベクトル \mathbf{g}_i の分割数 J 。

ベリー位相計算を行うために原子を変位させる必要があり、そのための機能が **Berry-Phonon** には備わっている。それは、次のように **displacement** ブロックを原子座標を指定する **atom_list** 内に記述することでできる。

```
atom_list{
  coordinate_system = cartesian
  atoms{
    #tag    rx          ry          rz          element
           0.000      0.000      0.000      Al
           2.6561175  2.6561175  2.6561175  As
  }
  displacement{
    sw_displace_atom = <ON_OFF>
    displaced_atom = <ATOM_ID>
    ux = <Ux>
    uy = <Uy>
    uz = <Uz>
  }
}
```

表 4. 原子を変位させる変数の説明

変数名またはタグ名	既定値	説明
sw_displace_atom	OFF	原子を変位されるかどうかのスイッチ。
displaced_atom	0	変位させる原子の番号。
ux	0.0	x 方向の原子変位量または、格子ベクトル \mathbf{a} に沿った内部座標の変化量。
uy	0.0	y 方向の原子変位量または、格子ベクトル \mathbf{b} に沿った内部座標の変化量。
uz	0.0	z 方向の原子変位量または、格子ベクトル \mathbf{c} に沿った内部座標の変化量。

本プログラムには有効電荷を計算するための機能が備わっている。それを行うには **Postprocessing** ブロック内に **Polarization** ブロックを加えて制御する。入力形式は 次のようになっている。

```
Postprocessing{
  Polarization{
    sw_bp_property = <ON_OFF>
    property = effective_charge
  }
}
```

表 5. 分極に関する物理量計算に関する変数の説明

変数名またはタグ名	既定値	説明
Polarization		結晶の分極に関する物性値の計算を制御するブロック。
sw_bp_property	OFF	Berry 位相分極に関連した物性値を計算するためのスイッチ
property	0	polarization を指定すると berry.data を読み込み Berry 位相分極を計算する。effective_charge を指定すると berry.data を読み込みボルン有効電荷を計算する。

Phonon ブロックの形式は次のようになっている。

```
Phonon{
  sw_phonon = <ON_OFF>
  sw_calc_force = <ON_OFF>
  displacement = <U>
  sw_vibrational_modes = <ON_OFF>
  point_group = <Point_Group>
  sw_lo_to_splitting = <ON_OFF>
  electronic_dielectric_constant{
    exx = 0.0, eyy = 0.0, ezz = 0.0,
    exy = 0.0, eyz = 0.0, ezx = 0.0
  }
  k_vector{ kx = 0.0, ky = 0.0, kz = 0.0 }
  sw_lattice_dielectric_tensor = <ON_OFF>
  sw_dielectric_function = <ON_OFF>
  energy_range{
    min_energy = 0.0
    max_energy = 0.01
    division_number = 100
  }
}
```

各変数またはタグの説明を表 6 と 7 にあげる。

表 6. 振動解析に関する変数の説明

変数名またはタグ名	既定値	説明
sw_phonon	OFF	格子振動解析設定ブロックを有効にするかどうかのスイッチ。
sw_calc_force	OFF	振動解析のための力計算を行うかどうかのスイッチ。 ON のときには、格子振動解析のための力計算を行う。(計算した力は force.data に出力される。) OFF のときには、sw_vibrational_modes=ON ならファイル"F_FORCE"から力のデータを読み込む。
displacement	0.1	原子変位パラメーター。
sw_vibrational_modes	OFF	格子振動解析を行うかどうかのスイッチ。 ON のときには、格子振動解析が行われ、modes.data に結果が出力される。 OFF のときには、格子振動解析は行われない。
point_group	C1	シェンフリース記号での点群の名称。
sw_lo_to_splitting	OFF	LO-TO 分裂を考慮するかどうかのスイッチ。
electronic_dielectric_constant		電子誘電率を指定するブロックのタグ名。
exx	0.0	電子誘電率のxx成分。
eyy	0.0	電子誘電率のyy成分。
ezz	0.0	電子誘電率のzz成分。
exy	0.0	電子誘電率のxy成分。
eyz	0.0	電子誘電率のyz成分。
ezx	0.0	電子誘電率のzx成分。
k_vector		格子振動の波数ベクトルの方向を指定するブロックのタグ。
kx	0.0	格子振動の波数ベクトルの x 成分。
ky	0.0	格子振動の波数ベクトルの y 成分。
kz	1.0	格子振動の波数ベクトルの z 成分。
sw_lattice_dielectric_tensor	OFF	格子誘電率を計算するかどうかのスイッチ。
sw_dielectric_function	OFF	誘電関数をファイル"F_EPSILON"に出力するかどうかのスイッチ。
energy_range		誘電関数のエネルギー範囲を指定するブロックのタグ。
min_energy	0.0	エネルギー範囲の最小値。
max_energy	0.01	エネルギー範囲の最大値。

division_number	100	エネルギー範囲の分割数。
-----------------	-----	--------------

変数 point_group で指定可能なシェンフリース記号を以下にあげる。

Oh, O, Td, Th, T, D4h, D4, D2d, C4v, C4h, S4, C4, D2h, D2, C2v, D6h, D6, D3h, C6v, C6h, C3h, C6, D3d, D3, C3v, S6, C3, C2h, Cs, C2, Ci, C1

11.4.1.3 入力および出力ファイル"F_BERRY"

原子 displaced_atom を(u_x, u_y, u_z)変位させて計算した場合に、ベリー位相データが次の形式で出力される。

```
nkprep, ig, displaced_atom, displacement(1:3)
do i=1,nkprep
  i, cphi(i), phi(i), wgh(i)
end do
```

配列 displacement の 1,2,3 番目の要素が u_x, u_y, u_z に対応する。 phi は逆格子ベクトル g_i に平行な線に沿ったベリー位相の配列であり、cphiはその phi の元となる 行列式の積の値の配列である。 nkprep は線積分の刻み J であり、ig は逆格子ベクトル g_i の指数であり、 wgh は k 点の重みである。

ベリー位相の入力形式は出力ファイルを結合したものであるが、ファイルの 先頭にはそのベリー位相データの数を記述しなければならない。 ただし、出力ファイルを結合する順序は問わない。

11.4.1.4 入力ファイル"F_EFFCHG"

ボルン有効電荷をベリー位相から計算せずに読み込むことができる。対称性から要求されるサイトのボルン有効電荷のみを記述すればよい。 次の形式で入力される。

```
num_zeff
do i=1,num_zeff
  ia
  zeff(1,1:3,ia)
  zeff(2,1:3,ia)
  zeff(3,1:3,ia)
end do
```

num_zeff はボルン有効電荷の数であり、ia が原子の番号であり、 zeff がボルン有効電荷の配列である。

11.4.1.5 入力および出力ファイル"F_FORCE"

"F_FORCE"には力の定数を計算するための力のデータが記述される。 その力データは次の形式で出力される。

```
num_force_data
do i = 1, num_force_data
  displaced_atom, displacement(1:3)
  do ia = 1, natm
    i, force_data(ia,1:3,i)
  end do
end do
```

num_force_data は力を計算する配置の数であり、displaced_atom は変位した原子 の番号であり、配列 displacement が原子の変位ベクトル(u_x, u_y, u_z)である。

11.4.1.6 出力ファイル"F_MODE"

"F_MODE"には振動解析の結果が記述される。 まず最初に基本並進ベクトル $\mathbf{a}_i = (a_{ix}, a_{iy}, a_{iz})$ が次の形式で記述される。

```
--- primitive lattice vectors ---
a 1x a 1y a 1z
```

```

a_2x a_2y a_2z
a_3x a_3y a_3z

```

次に原子の数 **natm** と各原子の座標(x_i, y_i, z_i)と質量 m_i とラベル **name(i)**が次の形式記述される。

```

--- Equilibrium position and mass of each atom---
Natom = natm
do i=1,natm
  i  x(i)  y(i)  z(i)  m(i)  name(i)
end do

```

次に振動解析の結果が次の形式で記述される。

```

--- Vibrational modes ---
Nmode= nmode  Natom= natm
do m = 1,nmode
  n=  m  representation(m)  acvtive(m)
  hbarW= omega_ha(m)  Ha = omega_ev(m)  eV;  nu= omega_nu(m)  cm^-1
  do i=1,natm
    i  vec(m,i,1)  vec(m,i,2)  vec(m,i,3)
  end do
end do

```

representation は既約表現の配列である。**active(m)**はラマン活性なモードであれば **R** になり、赤外活性なモードであれば **IR** となる。両活性であれば、**IR&R** となる。サイレントモードの場合には何も表示されない。**vec** は固有ベクトルの配列で、**omega_ha** は Hartree 単位での振動数であり、**omega_ev** は電子ボルト単位での振動数であり、**omega_nu** は波数である。

格子誘電率を計算した場合には、モード有効電荷が付加されて、次の形式で出力される。

```

--- Vibrational modes ---
Nmode= nmode  Natom= natm
do m = 1,nmode
  n=  m  character(m)  active(m)
  hbarW= omega_ha(m)  Ha = omega_ev(m)  eV;  nu= omega_nu(m)  cm^-1
  do i=1,natm
    i  vec(m,i,1)  vec(m,i,2)  vec(m,i,3)
  end do
  Mode effective charge and its average:
  Z=  z(m,1)  z(m,2)  z(m,3)  Ave.=  zave
end do

```

そして、最後に誘電率が次の形式で出力される。

```

--- Lattice and static dielectric tensors ---
[  elat_xx  elat_xy  elat_xz ]  [  e0_xx  e0_xy  e0_xz ]
[  elat_yx  elat_yy  elat_yz ]  [  e0_yx  e0_yy  e0_yz ]
[  elat_zx  elat_zy  elat_zz ]  [  e0_zx  e0_zy  e0_zz ]

```

elat が格子誘電率であり、**e0** が静的誘電率である。

11.4.1.7 出力ファイル"F_EPSILON"

"F_EPSILON"には誘電関数の値がつぎの形式で出力される。

```

Energy(eV)  E1xx  E1yy  E1zz  E1yz  E1zx  E1xy  E2xx  E2yy  E2zz  E2yz  E2zx  E2xy
do i=0,division_number
  energy(i)  e1xx(i)  e1yy(i)  e1zz(i)  e1yz(i)  e1zx(i)  e1xy(i)  e2xx(i)  e2yy(i)  e2zz(
i)  e2yz(i)  e2zx(i)  e2xy(i)
end do

```

energy は eV 単位のエネルギーの値である。**e1xx,e1yy,e1zz,e1yz,e1zx,e1xy** はそれぞれ誘電関数の実部の **xx,yy,zz,yz,zx,xy** 成分である。**e2xx,e2yy,e2zz,e2yz,e2zx,e2xy** はそれぞれ誘電関数の虚部の **xx,yy,zz,yz,zx,xy** 成分である。

11.4.1.8 ボルン有効電荷の出力

ボルン有効電荷は output000 に出力される。まず最初に、"F_BERRY"から読み込んだベリー一位相から計算したボルン有効電荷が次の形式で出力される。

```
--- Calculated electronic effective charges ---
do i=1,num_atom_inputed
      [      zel_xx(i)      zel_xy(i)      zel_xz(i) ]
Zel ( i) = [      zel_yx(i)      zel_yy(i)      zel_yz(i) ]
      [      zel_zx(i)      zel_zy(i)      zel_zz(i) ]
end do
```

num_atom_inputed は有効電荷が計算された原子の数である。zel_xx(i),zel_xy(i)などは原子 i のボルン有効電荷の電子からの寄与の xx,xy,...成分である。

電子からの寄与にイオンの価数を加えた結果が次の形式で出力される。

```
--- Calculated effective charges ---
do i=1,num_atom_inputed
      [      zeff_xx(i) zeff_xy(i) zeff_xz(i) ]
Zeff( i) = [      zeff_yx(i) zeff_yy(i) zeff_yz(i) ]
      [      zeff_zx(i) zeff_zy(i) zeff_zz(i) ]
end do
```

zeff_xx(i),zeff_xy(i)などは原子 i のボルン有効電荷の xx,xy,...成分である。

サイトの対称性を考慮して、対称化されたボルン有効電荷が次の形式で出力される。

```
--- Symmetrized effective charges ---
do i=1,num_atom_inputed
      [      zsym_xx(i) zsym_xy(i) zsym_xz(i) ]
Zsym( i) = [      zsym_yx(i) zsym_yy(i) zsym_yz(i) ]
      [      zsym_zx(i) zsym_zy(i) zsym_zz(i) ]
end do
```

zsym_xx(i),zsym_xy(i)などは原子 i の対称化されたボルン有効電荷の xx,xy,...成分である。

先の対称化されたボルン有効電荷から等価原子のボルン有効電荷を計算した結果が次の形式で出力される。

```
--- Effective charges of all atoms ---
do i=1,natm
      [      zeff_xx(i) zeff_xy(i) zeff_xz(i) ]
Zeff( i) = [      zeff_yx(i) zeff_yy(i) zeff_yz(i) ]
      [      zeff_zx(i) zeff_zy(i) zeff_zz(i) ]
end do
```

zeff_xx(i),zeff_xy(i)などは原子 i のボルン対称化された有効電荷の xx,xy,...成分である。

ボルン有効電荷の成分の平均値が次の形式で出力される。

```
--- Averaged effective charges ---
      [      zave_xx      zave_xy      zave_xz ]
Zave = [      zave_yx      zave_yy      zave_yz ]
      [      zave_zx      zave_zy      zave_zz ]
```

zave_xx,zave_xy,...などはボルン有効電荷の xx,xy,...成分の平均値である。

最後に、補正されたボルン有効電荷が次の形式で出力される。

```
--- Corrected effective charges ---
do i=1,natm
      [      zeff_xx(i) zeff_xy(i) zeff_xz(i) ]
Zeff( i) = [      zeff_yx(i) zeff_yy(i) zeff_yz(i) ]
      [      zeff_zx(i) zeff_zy(i) zeff_zz(i) ]
end do
```

zeff_xx(i),zeff_xy(i)などは原子 i の補正されたボルン有効電荷の xx,xy,...成分である。

11.4.1.9 入力ファイル"F_STRFRC"

入力ファイル"F_STRFRC"には次の形式で内部座標を固定して結晶を歪ませた時に原子に作用する力を記述する。

```
num_force_data
do i = 1, num_force_data
  index(i) strain(i)
  do ia = 1, natm
    i, force_data(ia,1:3,i)
  end do
end do
```

11.4.2 計算例：水晶(α -quartz)の格子誘電率計算

11.4.2.1 計算手順

水晶の格子誘電率の計算を例として、格子誘電率の計算の仕方を説明する。まず、格子誘電率計算の準備の手順を示す。

1. 計算したい格子定数において、構造最適化を行う。
2. nfdynm.dataの最後に書かれている最適構造でのPHASEの入力を作成する。
3. ベリー位相計算を行うディレクトリberryと振動解析を行うディレクトリphononを作成する。
4. ディレクトリberryには、Perlスクリプトprep_zeff.plが参照する、入力のテンプレートをディレクトリtemplate_berryとtemplate_scfに置く。入力テンプレートは2.で作成した入力を編集して作成する。
5. 2.で作成した入力を編集して、振動解析、有効電荷計算と格子誘電率計算に必要な記述がある入力を作成し、ディレクトリphononに置く。
6. ディレクトリberryで、prep_zeff.plを実行して、自動実行のためのPerlスクリプトexec_zeff.plと実際に使用する入力を作成し、exec_zeff.plを実行して有効電荷計算に必要なベリー位相を計算する。なお、prep_zeff.plはbinディレクトリーにある。
7. ディレクトリphononで振動解析を行い、最後にベリー位相を読み込んで格子誘電率を計算する。
8. 1.の構造最適化の仕方については、本マニュアル2章や3章などを見よ。2.では原子の座標をデカルト座標で入力するので、coordinate_systemをcartesianに設定する。以降の節で、3から7について、順に説明する。

11.4.2.2 ベリー位相計算

ベリー位相の計算はekcalを用いて行うので、ekcalに入力する自己無頓着な電子密度が必要であるので、その計算の入力テンプレートを示す。水晶はシリコン原子と酸素原子からなるので、自己無頓着場計算のfile_names.dataには、シリコンのポテンシャルpotentail.Siと酸素のポテンシャルpotential.Oの指定がある。これらのポテンシャルは二階層上に置かれていなければならない。

```
&fname
F_INP    = './nfinput.data'
F_POT(1) = '../..//potential.Si'
F_POT(2) = '../..//potential.O'
F_CHGT   = './nfchgt.data'
&end
```

ベリー位相の計算では、各原子を平衡位置からX,Y,Z方向にわずかに変位させたときの電子密度が必要である。その入力"F_INP"のテンプレートを以下に示す。Accuracy{}ブロック中のforce_convergence{}ブロックに含まれるmax_force=0.1e+3は、原子を平衡位置から微量ずらしたときに発生する力がある程度大きくてもそのまま原子を止めておくための予防的な設定である。

```
Control{
  condition = 0 ! {0|1|2|3}|{initial|continuation|fixed_charge|fixed_charge_continuation}
  cpumax = 24 hour ! {sec|min|hour|day}
```



```

max_iteration = 60000
}

accuracy{
  cke_wavefunctions = 36.00 rydberg ! cke_wf
  cke_chargedensity = 300.00 rydberg ! cke_cd
  num_bands = 32
  ksampling{
    method = monk ! {monk|mesh|file|directin|gamma}
    mesh{ nx=2, ny=2, nz=2}
  }
  smearing{
    method = parabolic ! {parabolic|tetrahedral}
    width = 0.002 hartree
  }
  xctype = ldapw91
  scf_convergence{
    delta_total_energy = 1.e-10 hartree
    succession = 3 !default value = 3
  }
  force_convergence{
    max_force = 0.1e+3
  }

  initial_wavefunctions = matrix_diagon !{random_numbers|matrix_diagon}
  matrix_diagon{
    cutoff_wf = 10.0 rydberg ! cke_wf
  }
}

structure{
  unit_cell_type=Bravais
  unit_cell{
    !#units bohr degree
    a = 9.2, b= 9.2, c= 10.12, alpha=90.0, beta=90.0, gamma=120.0
  }

  symmetry{
    tspace{
      system = h
      generators {
        !#tag rotation tx ty tz
        E 0 0 0
      }
    }
    sw_inversion = 0
  }
  magnetic_state = para !{para|af|ferro}

  atom_list{
    coordinate_system = cartesian ! {cartesian|internal}
    atoms{
      !#default mobile=no
      !#tag rx ry rz element
      -2.136349214 -3.700265381 3.373333567 Si
      4.272698428 0.000000000 0.000000000 Si
      -2.136349214 3.700265381 6.746667135 Si
      2.511045782 2.203258231 1.129569348 O
      0.652554710 -3.276258553 4.502902914 O
      -3.163600490 1.073000321 7.876236482 O
      2.511045781 -2.203258231 -1.129569348 O
      0.652554708 3.276258553 5.617097788 O
      -3.163600489 -1.073000321 2.243764219 O
    }
    displacement{
      sw_displace_atom = on
      displaced_atom = <ATOM_ID>
      ux = <Ux>
      uy = <Uy>
      uz = <Uz>
    }
  }
}

```

```

    }
  }
  element_list{ #units atomic_mass
    !#tag element atomicnumber zeta dev mass
      Si      14      0.00 3.5 28.0855
      O       8       0.00 2.0 15.9994
    }
  }

  wavefunction_solver{
    solvers{
      !#tag sol till_n dts dte itr var prec cmix submat
      MSD 5 0.2 0.2 1 * on 2 off
      lm+MSD 20 0.2 1.0 100 tanh on 2 on
      rmm2p -1 1.0 1.0 * * on 1 on
    }
    line_minimization{
      dt_lower_critical = 0.1
      dt_upper_critical = 3.0
    }
    rmm{
      imGSrmm = 1
      rr_Critical_Value = 1.e-15
      edelta_change_to_rmm = 1.0e-6 hartree
    }
  }

  charge_mixing{
    mixing_methods{
      !#tag no method rmxs rmxe itr var prec istr nbmix update
      1 simple 0.50 0.90 400 tanh on
      2 broyden2 0.30 0.30 100 * on 3 5 RENEW
    }
  }
}

```

Berry 位相計算のときには、**generators** には単位元のみを指定する。 **atoms** ブロックの後にある **displacement** ブロックで原子変位の指定をする。

```

displacement{
  sw_displace_atom = on
  displaced_atom = <ATOM_ID>
  ux = <Ux>
  uy = <Uy>
  uz = <Uz>
}

```

sw_displace_atom が ON に設定されていると、**displaced_atom** で指定した原子が(ux,uy,uz)方向に変位する。これらの変数には<ATOM_ID>,<Ux>,<Uy>,<Uz>を指定しておく。これらは、**prep_zeff.pl** によって置き換えられ、実際に使用する入力を作成される。以上で示した **file_names.data** と "**F_INP**"をディレクトリ **template_scf** に置く。

ベリー位相計算の **file_names.data** のテンプレートを以下に示す。

```

&fname
F_INP = './nfinput.data'
F_POT(1) = '../..//potential.Si'
F_POT(2) = '../..//pontetail.O'
F_CHGT = '../<SCF_DIR>/nfchgt.data'
&end

```

自己無傾着場計算の **file_names.data** との違いは、'**F_CHGT**'の指定です。同じ原子配置の自己無傾着場計算のディレクトリから電子密度を読み込むようにする。**prep_zeff.pl** が<SCF_DIR>を適切なディレクトリ名に置き換え、実際に使用する **file_names.data** を作成する。

ベリー位相計算の入力"**F_INP**"の例を以下に示す。

```

Control{
  condition = 2 ! {0|1|2|3}|{initial|continuation|fixed_charge|fixed_charge_continuation}
  cpumax = 24 hour ! {sec|min|hour|day}
}

```

```

max_iteration = 400001
}

accuracy{
  cke_wavefunctions = 36.00 rydberg ! cke_wf
  cke_chargedensity = 300.00 rydberg ! cke_cd
  num_bands = 24
  ksampling{
  }
  smearing{
    method = parabolic ! {parabolic|tetrahedral}
    width = 0.002 hartree
  }
  xctype = ldapw91
  ek_convergence{
    sw_eval_eig_diff = on
    succession = 3
    num_max_iteration = 200
    delta_eigenvalue = 1.0e-8
    num_extra_bands = 0
  }

  initial_wavefunctions = matrix_diagon !{random_numbers|matrix_diagon}
  matrix_diagon{
    cutoff_wf = 10.0 rydberg ! cke_wf
  }
}

structure{
  unit_cell_type=Bravais
  unit_cell{
    !#units bohr degree
    a = 9.2, b= 9.2, c= 10.12, alpha=90.0, beta=90.0, gamma=120.0
  }

  symmetry{
    tspace{
      system = h
      generators {
        !#tag rotation tx ty tz
        C3+ 0 0 2/3
        C212 0 0 0/1
      }
    }
    sw_inversion = 0
  }
  magnetic_state = para !{para|af|ferro}

  atom_list{
    coordinate_system = cartesian ! {cartesian|internal}
    atoms{
      !#default mobile=no
!#tag rx ry rz element
-2.136349214 -3.700265381 3.373333567 Si
4.272698428 0.000000000 0.000000000 Si
-2.136349214 3.700265381 6.746667135 Si
2.511045782 2.203258231 1.129569348 O
0.652554710 -3.276258553 4.502902914 O
-3.163600490 1.073000321 7.876236482 O
2.511045781 -2.203258231 -1.129569348 O
0.652554708 3.276258553 5.617097788 O
-3.163600489 -1.073000321 2.243764219 O
    }
    displacement{
      sw_displace_atom = on
      displaced_atom = <ATOM_ID>
      ux = <Ux>
      uy = <Uy>
      uz = <Uz>
    }
  }
}

```

```

}
element_list{ #units atomic mass
  !#tag element atomicnumber zeta dev mass
      Si      14          0.00 3.5 28.0855
      O       8           0.00 2.0 15.9994
}
}

wavefunction_solver{
  solvers {
    !#tag sol till_n dts dte itr prec submat
        MSD      10 1.0 1.0 1 on on
        lm+MSD    20 1.0 1.0 1 on on
        rmm2p     -1 1.0 1.0 1 on on
  }
  line_minimization{
    dt_lower_critical = 0.1
    dt_upper_critical = 3.0
  }
  rmm{
    imGSrmm = 1
    rr_Critical_Value = 1.e-15
    edelta_change_to_rmm = 1.0e-4 hartree
  }
}

Berry_phase{
  sw_berry_phase = on
  g_index = <G_INDEX>
  mesh{ n1 = <MESH_N1>, n2 = <MESH_N2>, J = <MESH_J> }
}

```

Berry 位相計算のときには、**generators** には単位元のみを指定する。 **displacement** ブロックは自己無頓着場計算の入力テンプレートと同じように記述する。Berry 位相計算を行うには、**Berry_phase** ブロックで、**sw_berry_phase** を ON に設定して、**g_index** と **mesh** ブロックの **n1,n2,J** を設定する。**g_index,n1,n2,J** には、<G_INDEX>、<MESH_N1>、<MESH_N2>、<MESH_J>を指定しておきます。**prep_zeff.pl** がこれらを適切な値に置き換え、実際に使用する入力を作成される。以上で作成した、**file_names.data** と **"F_INP"**をディレクトリ **template_berry** におく。

有効電荷を求めるためのベリー位相を計算する入力は、計算入力自動生成 Perl スクリプト **prep_zeff.pl** によって生成される。**prep_zeff.pl** を引数を付けずに実行すると、以下のように、実行するときに付加する引数のリストが表示される。

```
prep_zeff.pl DISPLACEMENT ATOM LIST MESH1 MESH2 MESH3
```

DISPLACEMENT には原子変位量を Bohr 単位で指定する。ここでは、0.05 とする。**ATOM_LIST** にはダブルクオーテーションで、有効電荷を計算したい原子の番号のリストを指定する。等価な原子の有効電荷は対称性で結び付けられるので、二番目のシリコン原子と 4 番目の酸素原子の有効電荷を計算すればいい。この場合は、**ATOM_LIST** を **"2 4"** とする。**MESH1,MESAH2,MESH3** には、k 空間メッシュのパラメーター **n1,n2,J** を **n1 n2 J** の並びで入力する。ここでは、**"2 2 10"** とする。ここで説明した引数の値で、**prep_zeff.pl** を実行する。

```
$ $PATH TO PHASE0/bin/prep_zeff.pl "0.05" "2 4" "2 2 10" "2 2 10" "2 2 10"
```

ベリー位相計算を逐次あるいは並列に実行する Perl スクリプト **exec_zeff.pl** とベリー位相計算に必要な入力ファイルが生成される。その入力ファイルはディレクトリ **berry_ai_ua_gβ** (**berry_a0_gβ**)や **scf_ai_ua** (**scf_a0**) に作成される。ただし、 $\alpha, \beta = 1, 2, 3$ である。 i は原子の番号を表し、 α は原子変位の方向を表し ($1 \rightarrow x, 2 \rightarrow y, 3 \rightarrow z$)、 β は逆格子ベクトルのインデックスを表す。ベリー位相計算はディレクトリ **berry_ai_ua_gβ** (**berry_a0_gβ**) で **ekcal** を実行することで行われる。この **ekcal** の実行に必要な電子密度はディレクトリ **scf_ai_ua** (**scf_a0**) で **phase** を実行することで得られる。ベリー位相計算の手続きをフローチャートにすると、図 11-4 のようになる。

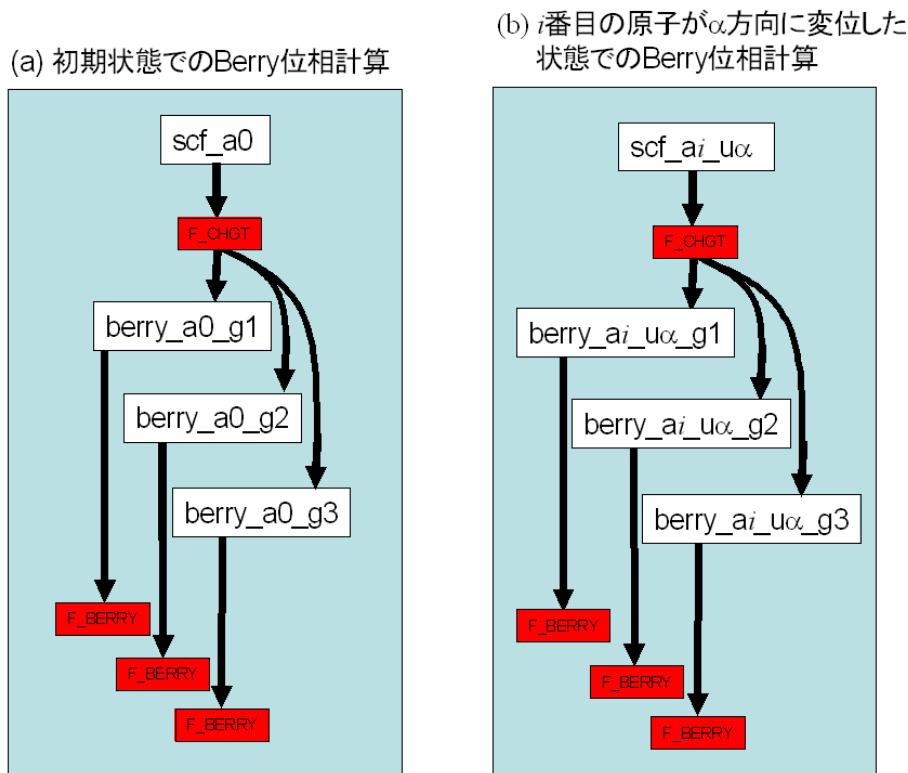


図 11-4 Berry 位相計算のフローチャート

Perl スクリプト `exec_zeff.pl` はこの手続きを自動で行ってくれる。`exec_zeff.pl` を引数を付けずに実行すると、以下のように実行するときに付加する引数のリストが表示される。

```
$ ./exec_zeff.pl PHASE EKCAL PARALLEL {-vpp|-primepower|-sr}
```

'PHASE'に PHASE のバイナリ `PATH_TO_PHASE0/bin/phase` を指定し、'EKCAL'に EKCAL のバイナリ `PATH_TO_PHASE0/bin/ekcal` を指定する。berry ディレクトリの下のディレクトリで `phase` と `ekcal` が実行されることを考慮して、実行環境に合うように設定する。'PARALLEL'には同時実行するプログラム(PHASE または EKCAL)の数を指定する。`exec_zeff.pl` を以下のように実行し、ベリー位相計算がすべて終了すると berry ディレクトリに `berry.data` が生成される。

```
$ ./exec_zeff.pl PATH_TO_PHASE0/bin/phase ¥
PATH_TO_PHASE0/bin/ekcal 1
```

MPI プログラム実行時にオプション `-machinefile` で利用可能なホストをを指定する場合には、`machinefile` を作成し、作業ディレクトリに置く。

VPP,PRIMEPOWER,SR8000,SR11000 といた大型計算機では MPI プログラムの実行方法が特殊であるので、MPI プログラムの実行方法を変更するオプション `-arch` がある。計算機と `-arch` オプションの値の対応を表 8 に示す。このオプションは

```
./exec_zeff.pl PATH_TO_PHASE0/bin/phase PATH_TO_PHASE0/bin/ekcal 1 -arch=primepower
```

のように最後に付加して用いる。

表 8. `exec_zeff.pl` のオプション

計算機	arch の値
VPP5000	vpp
PRIMEPOWER HPC500	primepower
SR11000	sr

バッチ処理システムによっては、使用する CPU リソースを制御することがある。`exe_zeff.pl` は最初とは異なる並列度で `mpirun` を実行することがある。このときに CPU リソースを制御されると正常に計算できない。このような場合に対応するために、`-loadleveler` というオプションがある。たとえば、SR11000 で 4 並列で実行するならば、以下のようにする。

```
./exec zeff.pl PATH TO PHASE0/bin/phase PATH TO PHASE0/bin/ekcal 4 -arch=sr -loadleveler
```

使用する CPU リソースが変更になった時点で `exec_zeff.pl` は終了する。 `exec_zeff.pl` のコメントに従い、使用する CPU リソースを減らして、`-loadleveler` オプションを外し、ジョブを投入する。

計算が途中で終了したときには、再度 `exec_zeff.pl` を実行することにより、継続計算ができる。

Perl が使えない環境では、図 3 に図示した順序で計算を実行する。その後、各 `berry` ディレクトリの `berry.data` を結合したファイルを作成する。このファイルの先頭行に結合したファイルの個数を 書き、`berry.data` という名前で保存する。

11.4.2.3 ベリー位相計算の出力

ベリー位相計算の出力"`F_BERRY`"はたとえば以下になる。

4	1	2	0.0	0.0	0.500000000000000D-01
1	0.63456226917922D+00	-0.20888574273791D-02	-0.32917965405901D-02	0.250000000000000D+00	
2	0.63432394408679D+00	-0.20934910658866D-02	-0.33003380756268D-02	0.250000000000000D+00	
3	0.63289711025371D+00	-0.19971191896918D-02	-0.31555090531068D-02	0.250000000000000D+00	
4	0.63151794421351D+00	-0.20390845515460D-02	-0.32288511895501D-02	0.250000000000000D+00	

これは二番目の Si が平衡位置から x 方向に 0.05 a.u. 変位した時のベリー位相のデータである。 `berry.data` は出力された "`F_BERRY`" を結合したファイルで、ファイルの先頭には結合したファイルの数である 21 が記述されている。

11.4.2.4 格子振動解析

`phonon` ディレクトリの上にシリコン原子のポテンシャル `potential.Si` と酸素原子のポテンシャル `potential.O` があるので、格子振動解析を行う時の `file_names.data` は以下になる。

```
&fnames
F_INP      = './nfinput.data'
F_POT(1)   = '../potential.Si'
F_POT(2)   = '../potential.O'
&end
```

格子振動解析を行う時の入力 "`F_INP`" をつぎに示す。

```
Control{
  condition = 0 ! {0|1|2|3}|{initial|continuation|fixed_charge|fixed_charge_continuation}
  cpmmax = 24 hour ! {sec|min|hour|day}
  max_iteration = 60000
}

accuracy{
  cke_wavefunctions = 36.00 rydberg ! cke_wf
  cke_chargedensity = 300.00 rydberg ! cke_cd
  num_bands = 32
  ksampling{
    method = monk ! {monk|mesh|file|directin|gamma}
    mesh{ nx=2, ny=2, nz=2}
  }
  smearing{
    method = parabolic ! {parabolic|tetrahedral}
    width = 0.002 hartree
  }
}

xctype = ldapw91
scf_convergence{
  delta_total_energy = 1.e-10 hartree
  succession = 3 !default value = 3
}

initial_wavefunctions = matrix_diagon !{random_numbers|matrix_diagon}
matrix_diagon{
```

```

    cutoff_wf = 10.0 rydberg ! cke_wf
  }
}

structure{
  unit_cell_type=Bravais
  unit_cell{
    !#units bohr degree
    a = 9.2, b= 9.2, c= 10.12, alpha=90.0, beta=90.0, gamma=120.0
  }

  symmetry{
    tspace{
      system = h
      generators {
        !#tag rotation tx ty tz
        C3+      0  0  2/3
        C212     0  0  0/1
      }
    }
    sw_inversion = 0
  }
  magnetic_state = para !{para|af|ferro}

  atom_list{
    coordinate_system = cartesian ! {cartesian|internal}
    atoms{
      !#default mobile=no
!#tag rx ry rz element
      -2.136349214 -3.700265381 3.373333567 Si
      4.272698428 0.000000000 0.000000000 Si
      -2.136349214 3.700265381 6.746667135 Si
      2.511045782 2.203258231 1.129569348 O
      0.652554710 -3.276258553 4.502902914 O
      -3.163600490 1.073000321 7.876236482 O
      2.511045781 -2.203258231 -1.129569348 O
      0.652554708 3.276258553 5.617097788 O
      -3.163600489 -1.073000321 2.243764219 O
    }
    displacement{
      sw_displace_atom = off
      displaced_atom = 0
      ux = 0
      uy = 0
      uz = 0
    }
  }
  element_list{ #units atomic mass
    !#tag element atomicnumber zeta dev mass
    Si 14 0.00 3.5 28.0855
    O 8 0.00 2.0 15.9994
  }
}

wavefunction_solver{
  solvers{
    !#tag sol till_n dts dte itr var prec cmix submat
    MSD 5 0.2 0.2 1 * on 2 off
    lm+MSD 20 0.2 1.0 100 tanh on 2 on
    rmm2p -1 1.0 1.0 * * on 1 on
  }

  line_minimization{
    dt_lower_critical = 0.1
    dt_upper_critical = 3.0
  }

  rmm{
    imGSrmm = 1
    rr Critical Value = 1.e-15
  }
}

```

```

    edelta_change_to_rmm = 1.0e-6 hartree
}

}

charge_mixing{
  mixing_methods{
    !#tag no    method    rmxs    rmxe    itr    var    prec istr    nbmix    update
        1    simple    0.50    0.90    400    tanh    on      3    5    RENEW
        2    broyden2    0.30    0.30    100    *      on      3    5    RENEW
  }
  charge_preconditioning{
    amix = 0.90
    bmix = -1.00
  }
}

Phonon{
  sw_phonon = on
  sw_calc_force = on
  force_calc{
    start = 1, end = 0
  }
  displacement = 0.05
  norder = 1
  sw_polynomial_fit = on
  sw_vibrational_modes = on
  point_group = D3
  electronic_dielectric_constant{
    exx = 2.56011,    eyy = 2.56011,    ezz = 2.57411
    exy = 0.0,       eyz = 0.0,       ezx = 0.0
  }
  k_vector{ kx = 0.0, ky = 0.0, kz = 0.0 }
  sw_lattice_dielectric_tensor = off
}

Postprocessing{
  Polarization{
    sw_bp_property = off
    property = effective_charge
  }
}
}

```

displacement ブロックがあるときは、sw_diplace_atom が off になっていることを確認する。振動解析を行うので、element リストに質量(mass)指定が正しく行われていることを確認する。ここで、示した入力では、「#units atomic_mass」を element_list の先頭に記述し、原子質量単位で質量を入力している。Phonon ブロックに記述される振動解析に関わる入力の詳細は、5.2.1 に記載されている。sw_phonon と sw_calc_force を ON に指定し、原子変位(displacement)を 0.05 に指定する。水晶の結晶点群は D_3 なので、point_group に D3 を設定する。静的誘電率は格子誘電率と電子誘電率の和なので、electronic_dielectric_constant ブロックで、水晶の電子誘電率の計算値 $\epsilon_{xx} = \epsilon_{yy} = 2.56011$, $\epsilon_{zz} = 2.57411$ を指定している。まず、振動解析のみを行うので、sw_lattice_dielectric_tensor を off に設定する。この入力で振動解析を行った後に、sw_calc_force を off に、sw_lattice_dielectric_tensor と sw_bp_property を on に設定する。振動解析を行ったディレクトリでもう一度 PHASE を実行して、格子誘電率を計算する。こうして得られた PHASE の出力の見方を次節で説明する。

11.4.2.5 ボルン有効電荷の出力

output001 に出力される有効電荷計算の結果を以下に示す。

```
--- Calculated electronic effective charges ---
Zel ( 2) = [ -1.00158  0.00284 -0.00266 ]
           [ -0.00001 -0.35464 -0.32379 ]
           [  0.00000  0.28121 -0.56029 ]

Zel ( 4) = [ -7.31546  0.50495  0.31818 ]
           [  0.45145 -8.00477 -0.70090 ]
           [  0.25220 -0.73456 -7.72450 ]
```

タイトル”Calculated electronic effective charges”の後には、ベリ一位相から計算した電子からの有効電荷への寄与が 3x3 の行列として出力されている。’Zel’の後の括弧の中の数字は、原子の番号である。ボルン有効電荷を計算する原子を 2 番目と 4 番目の原子に指定した通りになっている。

```
--- Calculated effective charges ---
Zeff( 2) = [  2.99842  0.00284 -0.00266 ]
           [ -0.00001  3.64536 -0.32379 ]
           [  0.00000  0.28121  3.43971 ]

Zeff( 4) = [ -1.31546  0.50495  0.31818 ]
           [  0.45145 -2.00477 -0.70090 ]
           [  0.25220 -0.73456 -1.72450 ]
```

タイトル”Calculated effective charges”の後には、イオンの電荷を加え、ボルン有効電荷が出力されている。Zeff の後の括弧の中の数字は Zel と同じ意味である。

```
--- Symmetrized effective charges ---
Zsym( 2) = [  2.99842  0.00000  0.00000 ]
           [  0.00000  3.64536 -0.32379 ]
           [  0.00000  0.28121  3.43971 ]

Zsym( 4) = [ -1.31546  0.50495  0.31818 ]
           [  0.45145 -2.00477 -0.70090 ]
           [  0.25220 -0.73456 -1.72450 ]
```

タイトル”Symmetrized effective charges”の後には、原子サイトの対称性を満たすようにしたボルン有効電荷が出力されている。

```
--- Effective charges of all atoms ---
Zeff( 1) = [  3.48362 -0.28013 -0.28041 ]
           [ -0.28013  3.16016  0.16190 ]
           [  0.24354 -0.14061  3.43971 ]

Zeff( 2) = [  2.99842  0.00000  0.00000 ]
           [  0.00000  3.64536 -0.32379 ]
           [  0.00000  0.28121  3.43971 ]

Zeff( 3) = [  3.48362  0.28013  0.28041 ]
           [  0.28013  3.16016  0.16190 ]
           [ -0.24354 -0.14061  3.43971 ]

Zeff( 4) = [ -1.31546  0.50495  0.31818 ]
           [  0.45145 -2.00477 -0.70090 ]
           [  0.25220 -0.73456 -1.72450 ]

Zeff( 5) = [ -2.24657  0.08613 -0.76609 ]
           [  0.03263 -1.07365  0.07490 ]
           [ -0.76225  0.14887 -1.72450 ]

           [ -1.41830 -0.51083  0.44791 ]
```

```

Zeff( 6) = [ -0.56433 -1.90192  0.62600 ]
           [  0.51005  0.58569 -1.72450 ]

           [ -1.31546 -0.50495 -0.31818 ]
Zeff( 7) = [ -0.45145 -2.00477 -0.70090 ]
           [ -0.25220 -0.73456 -1.72450 ]

           [ -2.24657 -0.08613  0.76609 ]
Zeff( 8) = [ -0.03263 -1.07365  0.07490 ]
           [  0.76225  0.14887 -1.72450 ]

           [ -1.41830  0.51083 -0.44791 ]
Zeff( 9) = [  0.56433 -1.90192  0.62600 ]
           [ -0.51005  0.58569 -1.72450 ]

```

タイトル”Effective charges of all atoms”の後には、計算しなかった等価原子のボルン有効電荷が結晶の対称性を用いて構成され、すべての原子のボルン有効電荷が出力されている。

```

--- Averaged effective charges ---
      [  0.00055  0.00000  0.00000 ]
Zave = [  0.00000  0.00055  0.00000 ]
      [  0.00000  0.00000 -0.00310 ]

```

タイトル”Averaged effective charges”の後には、ボルン有効電荷の平均値が出力されている。結晶を構成する原子のボルン有効電荷を足し合わせると、ゼロにならなければならない。ボルン有効電荷の平均値を各原子のボルン有効電荷から引くことによりその性質を満たすように修正することができる。修正されたボルン有効電荷はタイトル”Corrected effective charges”以下に出力されている。

```

--- Corrected effective charges ---
      [  3.48307 -0.28013 -0.28041 ]
Zeff( 1) = [ -0.28013  3.15960  0.16190 ]
           [  0.24354 -0.14061  3.44281 ]

           [  2.99787  0.00000  0.00000 ]
Zeff( 2) = [  0.00000  3.64480 -0.32379 ]
           [  0.00000  0.28121  3.44281 ]

           [  3.48307  0.28013  0.28041 ]
Zeff( 3) = [  0.28013  3.15960  0.16190 ]
           [ -0.24354 -0.14061  3.44281 ]

           [ -1.31602  0.50495  0.31818 ]
Zeff( 4) = [  0.45145 -2.00532 -0.70090 ]
           [  0.25220 -0.73456 -1.72141 ]

           [ -2.24713  0.08613 -0.76609 ]
Zeff( 5) = [  0.03263 -1.07421  0.07490 ]
           [ -0.76225  0.14887 -1.72141 ]

           [ -1.41886 -0.51083  0.44791 ]
Zeff( 6) = [ -0.56433 -1.90248  0.62600 ]
           [  0.51005  0.58569 -1.72141 ]

           [ -1.31602 -0.50495 -0.31818 ]
Zeff( 7) = [ -0.45145 -2.00532 -0.70090 ]
           [ -0.25220 -0.73456 -1.72141 ]

           [ -2.24713 -0.08613  0.76609 ]
Zeff( 8) = [ -0.03263 -1.07421  0.07490 ]
           [  0.76225  0.14887 -1.72141 ]

           [ -1.41886  0.51083 -0.44791 ]

```

```

Zeff( 9) = [ 0.56433 -1.90248 0.62600 ]
            [ -0.51005 0.58569 -1.72141 ]

```

11.4.2.6 格子振動解析と誘電率の出力

出力ファイル"F_MODE"を次に示す。

```

--- primitive lattice vectors ---
 9.20000000000 0.0000000000 0.0000000000
-4.60000000000 7.9674337148 0.0000000000
 0.00000000000 0.0000000000 10.1200000000
--- Equilibrium position and mass of each atom---
Natom= 9
 1 -2.1363492140 -3.7002653810 3.3733335670 51196.42133 Si
 2 4.2726984280 0.0000000000 0.0000000000 51196.42133 Si
 3 -2.1363492140 3.7002653810 6.7466671350 51196.42133 Si
 4 2.5110457820 2.2032582310 1.1295693480 29164.94360 O
 5 0.6525547100 -3.2762585530 4.5029029140 29164.94360 O
 6 -3.1636004900 1.0730003210 7.8762364820 29164.94360 O
 7 2.5110457810 -2.2032582310 -1.1295693480 29164.94360 O
 8 0.6525547080 3.2762585530 5.6170977880 29164.94360 O
 9 -3.1636004890 -1.0730003210 2.2437642190 29164.94360 O
--- Vibrational modes ---
Nmode= 27 Natom= 9
n= 1 E IR&R
 hbarW= 0.00000000E+00 Ha = 0.00000000E+00 eV; nu= 0.00000000E+00 cm^-1
...
n= 4 E IR&R
 hbarW= 0.58285132E-03 Ha = 0.15860191E-01 eV; nu= 0.12792108E+03 cm^-1
 1 -0.1076861591 -0.0492955392 -0.0289969411
 2 -0.0000000001 0.1372246738 0.0579965405
 3 0.1076861592 -0.0492955391 -0.0289969411
 4 -0.0710674524 0.2294802586 -0.2767253313
 5 -0.0266667306 -0.2719065772 -0.0591536507
 6 0.3853480075 0.0172331774 0.3358772210
 7 0.0710674522 0.2294802589 -0.2767253316
 8 0.0266667309 -0.2719065770 -0.0591536503
 9 -0.3853480076 0.0172331772 0.3358772209
Mode effective charge and its norm:
Z= 0.0000000000 0.0159080661 0.0000020096 Norm= 0.0159080663
n= 5 E IR&R
 hbarW= 0.58285208E-03 Ha = 0.15860211E-01 eV; nu= 0.12792125E+03 cm^-1
 1 0.0750340743 -0.1076820548 0.0502260531
 2 -0.1114922066 -0.0000000001 0.0000000000
 3 0.0750340742 0.1076820548 -0.0502260531
 4 -0.2462863619 -0.2627994415 0.2280661351
 5 0.2551052896 -0.2184164183 -0.3536922309
 6 -0.0340512209 0.1936020463 0.1256104976
 7 -0.2462863620 0.2627994412 -0.2280661348
 8 0.2551052896 0.2184164186 0.3536922309
 9 -0.0340512205 -0.1936020463 -0.1256104979
Mode effective charge and its norm:
Z= 0.0159051835 0.0000000000 0.0000000000 Norm= 0.0159051835
...
--- Lattice and static dielectric tensors ---
[ 2.1944 0.0000 0.0000 ] [ 4.7545 0.0000 0.0000 ]
[ 0.0000 2.1944 0.0000 ] [ 0.0000 4.7545 0.0000 ]
[ 0.0000 0.0000 2.3874 ] [ 0.0000 0.0000 4.9616 ]

```

各モードの振動数と固有ベクトルの次にモード有効電荷が出力されている。'Z='の次に並んでいる三つの値がモード有効電荷のデカルト座標での三成分である。'Norm='の次の値はこのベクトルの大きさである。最後の三行に格子誘電率が出力されている。右側の 3x3 の行列が格子誘電率である。左側にはその格子誘電率に電子誘電率を加えた行列が出力されている。これから、格子誘電率は $\epsilon_{xx}^{lat} = \epsilon_{yy}^{lat} = 2.2, \epsilon_{zz}^{lat} = 2.4$ と計算されたことが分かる。そして、静的誘電率が $\epsilon_{xx}^0 = \epsilon_{yy}^0 = 4.8, \epsilon_{zz}^0 = 5.0$ となることが分かる。

11.4.3 計算例：AlN の圧電定数の計算

11.4.3.1 計算手順

AlN の圧電定数の計算を例として、圧電定数の計算の仕方を説明する。まず、圧電定数計算の準備の手順を示す。

1. 理論的な平衡格子定数を決定する。
2. 平衡格子定数において、構造最適化を行う。
3. nfdynm.dataの最後に書かれている最適構造でのPHASEの入力を作成する。
4. ベリー位相計算を行うディレクトリberryを作成する。
5. ディレクトリberryには、Perlスクリプトprep_zeff.plが参照する、入力のテンプレートをディレクトリtemplate_berryとtemplate_scfiに置く。入力テンプレートは3.で作成した入力を編集して作成する。
6. 振動解析を行うディレクトリphononを作成する。
7. 3.で作成した入力を編集して、振動解析、有効電荷計算と圧電応答解析に必要な記述がある入力を作成し、ディレクトリphononに置く。
8. 圧電応答解析を行うディレクトリpiezoを作成し、その下にディレクトリclampedとinternalを作成する。
9. ディレクトリclampedには、Perlスクリプトprep_piezo.plが参照する、入力のテンプレートをディレクトリtemplate_berryとtemplate_scfiに置く。入力テンプレートは3.で作成した入力を編集して作成する。
10. ディレクトリinternalには、Perlスクリプトprep_strfrc.plが参照する、入力のテンプレートをtemplate_scfiに置く。入力テンプレートは3.で作成した入力を編集して作成する。
11. ディレクトリberryで、prep_zeff.plを実行して、自動実行のためのPerlスクリプトexec_zeff.plと実際に使用する入力を作成し、exec_zeff.plを実行して有効電荷計算に必要なベリー位相を計算する。なお、prep_zeff.plはPATH_TO_PHASE0/binにある。
12. ディレクトリphononで振動解析のための力計算を行う。
13. ディレクトリpiezo/clampedで、prep_piezo.plを実行して、自動実行のためのPerlスクリプトexec_piezo.plと実際に使用する入力を作成し、exec_piezo.plを実行してイオン固定圧電定数の計算に必要なベリー位相を計算する。なお、prep_piezo.plはPATH_TO_PHASE0/binにある。
14. ディレクトリpiezo/internalで、prep_strfrc.plを実行して、自動実行のためのPerlスクリプトexec_strfrc.plと実際に使用する入力を作成し、exec_strfrc.plを実行してひずみ-力結合定数の計算に必要な原子に作用する力を計算する。なお、prep_strfrc.plはPATH_TO_PHASE0/binにある。
15. ディレクトリphononで振動解析を行い、最後にベリー位相とひずみ-力結合定数を読み込んで圧電定数を計算する。

1.格子定数決定と 2.構造最適化の仕方については、2 章や 3 章を見よ。3.では原子の座標をデカルト座標で入力するので、coordinate_system を cartesian に設定する。以降の節で、4 から 15 について、順に説明する。

11.4.3.2 ベリー位相計算

入力の作り方と計算方法は格子誘電率計算の場合と同一です。入力例がディレクトリ samples/lattice/AlN/berry にあります。AlN の場合、prep_zeff.pl は次のように実行すれば、正しい結果が得られる入力を作成できます。

```
prep_zeff.pl 0.1 '1 3' '6 6 15' '6 6 15' '6 6 15'
```

原子変位を 0.1 a.u.とし、1 番目と 3 番目の原子のボルン有効電荷を実際に計算するように指定しています。n1xn2xJ はすべての方向で 6x6x15 としています。あとは、作成された exec_zeff.pl を次のように実行してください。

```
exec_zeff.pl phase ekcal 1
```

最後の 1 は 1 MPI 並列で計算することを指定しています。この入力では最大 7 MPI 並列まで指定できます。

11.4.3.3 格子振動解析

入力の作り方と計算方法は格子誘電率計算の場合と同一です。入力例がディレクトリ `samples/lattice/AlN/phonon` にあります。この計算が完了した段階で格子誘電率を計算することができます。

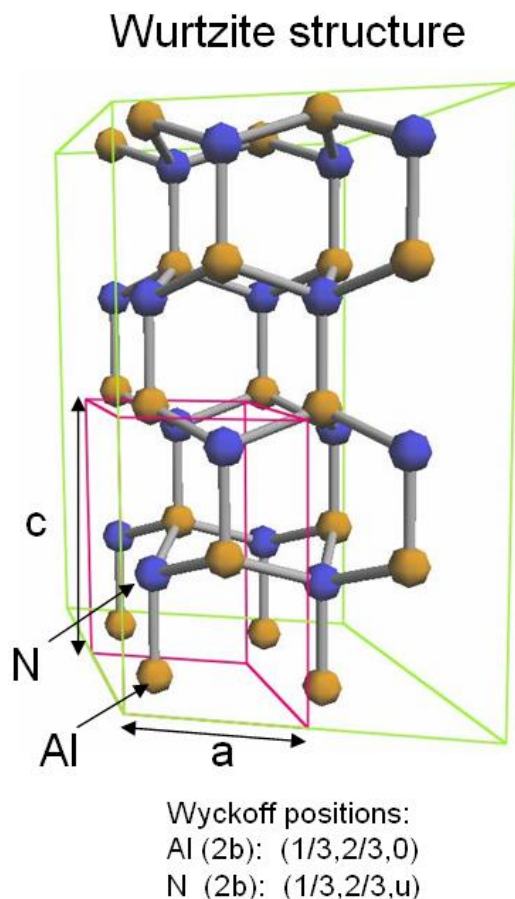


図 11-5 Wurtzite 構造の AlN

11.4.3.4 圧電定数のイオン固定項の計算

ディレクトリ `berry` に作成した入力テンプレートに類似したものをディレクトリ `piezo/clamped` に作成します。原子の座標は内部座標で入力します。図 4 に示す構造の AlN の場合、次のようになります。

```
atom_list{
  coordinate_system = internal ! {cartesian|internal}
  atoms{
    !#tag    rx          ry          rz          element mobile
    0.333333333 0.666666666 0.000000000 Al      off
    0.666666666 0.333333333 0.500000000 Al      off
    0.333333333 0.666666666 0.382000000 N       on
    0.666666666 0.333333333 0.882000000 N       on
  }
}
```

ボルン有効電荷の計算ではないので、原子を変位させる入力はありません。圧電定数を計算する場合は、結晶を歪ませるための以下に示す入力が必要になります。

```
strain{
  sw_strained_cell = ON
}
```

```

    e11 = <E11>, e22 = <E22>, e33 = <E33>
    e23 = <E23>, e32 = <E32>
    e31 = <E31>, e13 = <E13>
    e12 = <E12>, e21 = <E21>
}

```

<E11>などは `prep_piezo.pl` によって置き換えられて、実際に使用される入力を作成されます。ベリー一位相計算の入力はボルン有効電荷の計算の場合と同じです。

```

Berry_phase{
    sw_berry_phase = on
    g_index = <G_INDEX>
    mesh{ n1 = <MESH_N1>, n2 = <MESH_N2>, J = <MESH_J> }
}

```

`prep_piezo.pl` が読み込む入力テンプレートが作成したら、`prep_piezo.pl` を実行します。引数を付けずに `prep_piezo.pl` を実行すると、引数の情報が得られます。

```
prep_piezo.pl STRAIN INDEX_LIST MESH1 MESH2 MESH3
```

STRAIN はひずみの大きさで、**INDEX_LIST** は零でないひずみ成分の縮約表示の指数のリストです。**MESH1** などは `prep_zeff.pl` と同じです。**AIN** の場合、次のようにすれば圧電定数の零でないすべての成分 e_{31} , e_{33} , e_{15} を計算できます。

```
prep_piezo.pl 0.01 '1 3 5' '6 6 15' '6 6 15' '6 6 15'
```

ひずみの大きさを 0.01 として、零でないひずみ成分を $\epsilon_1, \epsilon_3, \epsilon_5$ としています。生成された `exec_piezo.pl` を次のように実行します。

```
exec_piezo.pl phase ekcal 1
```

最後の 1 は 1 MPI 並列で計算することを指定しています。この入力では最大 4MPI 並列まで指定できます。

計算が完了すると `berry.data` が `piezo/clamped` ディレクトリに作成されます。それを `scf_e0` ディレクトリにコピーして、入力 `nfinput.data` に

```

Postprocessing{
    polarization{
        sw_bp_property = ON
        property = piezoelectric_const
    }
}

```

を書き加えます。`condition` を以下のように書き変えて、継続計算を行います。

```

Control{
    condition = continuation
}

```

出力 `output001` に次のような圧電定数のイオン固定項の値が出力されます。

```

=== Piezoelectric constant (C/m^2) ===
 1    -0.0000488692    0.0000008508    0.2508631403
 2     0.0000000000    0.0000000000    0.0000000000
 3     0.0000015381    0.0000020659   -0.3977128777
 4     0.0000000000    0.0000000000    0.0000000000
 5     0.3321219558    0.0000004282    0.0030073377
 6     0.0000000000    0.0000000000    0.0000000000

```

列は左から x, y, z 方向に対応してます。この結果は圧電定数のイオン固定項の値が $e_{31}^{(0)} = 0.251 \text{C/m}^2$, $e_{33}^{(0)} = -0.398 \text{C/m}^2$, $e_{15}^{(0)} = 0.332 \text{C/m}^2$ であることを示しています。零であるべきところが零になっていませんが、それは数値計算誤差によるものです。

11.4.3.5 圧電定数の内部ひずみ項の計算

prep_strfrc.pl が読み込む入力テンプレート template_scf はディレクトリ clamped に作成したものをコピーして作成してください。

```
cd piezo/internal
```

```
cp -R ../clamped/template_scf .
```

prep_strfrc.pl を引数を付けずに実行すると、引数の情報が得られます。

```
prep_strfrc.pl STRAIN INDEX LIST
```

引数の意味は prep_piezo.pl の引数と同じです。AlN の場合、次のように実行します。

```
prep_strfrc.pl 0.02 '1 3 5'
```

ひずみ-力結合定数を差分で計算するとき、大きさの同じ正と負のひずみで、結晶を変形させます。そのひずみの大きさを 0.02 にしています。生成された exec_strfrc.pl を次のように実行します。

```
exec_strfrc.pl phase 1
```

最後の 1 は 1 MPI 並列で計算することを指定しています。この入力では最大 6MPI 並列まで指定できます。

計算が完了すると、ディレクトリ piezo/internal に strfrc.data が作成されます。これをディレクトリ phonon にコピーします。ディレクトリ phonon に移り、入力 nfinput.data を次のように書き換えます。

```
Phonon{
  sw_phonon = on
  sw_calc_force = off
  displacement = 0.05
  sw_vibrational_modes = on
  sw_internal_strain_piezoelectric_tensor = on
}
Postprocessing{
  Polarization{
    sw_bp_property = on
    property = effective_charge
  }
}
```

sw_calc_force を off とし、sw_internal_strain_piezoelectric_tensor を on とし、sw_bp_property を on とします。

phase を実行して得られる出力 output001 に次のような圧電定数の内部ひずみ項の値が出力されます。

```
=== Internal-strain piezoelectric tensor (C/m^2) ===
 1      0.0000000000      -0.0001106316      -0.9017585866
 2      0.0000000000      0.0000000000      0.0000000000
 3      0.0000000000      0.0000000000      2.0047352963
 4      0.0000000000      0.0000000000      0.0000000000
 5     -0.6775051139      0.0000000000      0.0000006994
 6      0.0000000000      0.0000000000      0.0000000000
```

列は左から x,y,z 方向に対応してます。この結果は圧電定数の内部ひずみ項の値 $e_{ij}^{(1)}$ が $e_{31}^{(1)} = -0.902C/m^2$,

$e_{33}^{(1)} = 2.005C/m^2$, $e_{15}^{(1)} = -0.678C/m^2$ であることを示しています。零であるべきところが零になっていませ

んが、それは数値計算誤差によるものです。

AlN の圧電定数の計算結果を表 9 にまとめて示します。計算結果は実験と一致しています。

表 11.2 AlN の圧電定数(C/m^2)

成分	イオン固定項	内部ひずみ項	合計	実験値
e_{31}	0.251	-0.902	-0.651	-0.58
e_{33}	-0.398	2.005	1.607	1.55
e_{15}	0.332	-0.678	-0.345	-0.48

11.4.4 計算例：GaAs-格子誘電率計算

11.4.4.1 ベリー位相計算

ベリー位相計算はekcalを用いて行います。

まず、GaAs 結晶のボルン有効電荷計算を求めるのに必要とされるベリー位相の計算を行うために、samples/lattice/GaAs/berry というディレクトリに移ります。

```
cd PATH_TO_PHASE0/samples/GaAs/berry
```

ここには、ベリー位相計算の入力テンプレートが収められている template_berry とベリー位相計算の入力として与える電子密度を計算する SCF 計算の入力テンプレートが template_scf に収められています。これらのファイルは実際に使用する入力を作成する Perl スクリプト prep_zeff.pl が参照します。まず、template_scf にある nfinput.data には通常の SCF 計算の入力と違い、原子を X,Y,Z 方向に変位させるための、記述があります。それは、atoms ブロックの次に書かれています。

```
displacement{
  sw_displace_atom = on
  displaced_atom = <ATOM_ID>
  ux = <Ux>
  uy = <Uy>
  uz = <Uz>
}
```

displaced_atom には変位させる原子を指定し、原子の変位ベクトルがデカルト座標のベクトル(ux,uy,uz)と表すものとして、ux,uy,uz に入力します。(ただし、coordinate_system は cartesian になっているものとします。)

displaced_atom,ux,uy,uz には<ATOM_ID>,<Ux>,<Uy>,<Uz>が指定されていますが、これらは calc_zeff.sh が実際に使用する入力を作るときに置き換えますので、必ずこのように入力してください。

template_berry に収められている nfinput.data を見ると、ベリー位相計算を行うときに必要となるタグが見つかります。

```
Berry_phase{
  sw_berry_phase = on
  g_index = <G_INDEX>
  mesh{ n1 = <MESH_N1>, n2 = <MESH_N2>, J = <MESH_J> }
}
```

g_index には逆格子ベクトルのインデックス(1、2、または3)を指定します。ブロック mesh では、k 空間のメッシュを n1,n2,J で指定します。n1,n2 は scf 計算の時と同程度にします。J はそれよりも3倍から5倍大きくとります。g_index,n1,n2,J には<G_INDEX>,<MESH_N1>,<MESH_N2>,<MESH_J>が指定されていますが、これらは prep_zeff.pl が実際に使用する入力を作るときに置き換えますので、必ずこのように入力してください。ディレクトリに PATH_TO_PHASE0/bin ある prep_zeff.pl を以下のように実行します。

```
$ PATH_TO_PHASE0/bin/prep_zeff.pl 0.05 '1 2' '4 4 20' '4 4 20' '4 4 20'
```

一番目の引数が原子変位量であり、二番目の引数がボルン有効電荷を計算する原子の番号のリストです。最後の三つの引数では、'n1 n2 J'の様にメッシュパラメータ(n1,n2,J)を指定します。最初のメッシュパラメータは一番目の逆格子ベクトルに対して適用されます。つぎは二番目、その次は三番目というように対応付けられています。上記の様に実行すると、原子変位量は0.05 bohr になり、1 番目の原子(Ga)と2 番目の原子(As)のボルン有効電荷を計算します。どの逆格子ベクトルに対しても(4,4,20)メッシュパラメーターが適用されます。この Perl スクリプトは template_scf と template_berry を参照して、実際に使用する入力を生成します。実際に使用する入力は scf_a0,...,berry_a0_g1,... といったディレクトリに置かれます。これらのディレクトリで phase または ekcal を実行しなければなりません、それを行う Perl スクリプト exec_zeff.pl が生成されていますので、それを以下のようにして実行します。

```
$ ./exec_zeff.pl "PATH_TO_PHASE0/bin/phase" ¥
```



```
" PATH_TO_PHASE0/bin/ekcal" "1"
```

このスクリプトの実行が終了すると、**berry.data** というファイルが生成されます。これには、ボルン有効電荷を計算するためのベリー位相のデータが収められています。これを **phonon** ディレクトリにコピーします。

```
$ cp berry.data ../phonon
```

11.4.4.2 振動解析および格子誘電率計算

次に、振動解析を行うディレクトリ **phonon** に移ります。

```
$ cd ../phonon
```

このディレクトリの **PHASE** の入力 **nfinput.data** には振動解析を制御する **Phonon** ブロックがあります。

```
Phonon{
  sw_phonon = on
  sw_calc_force = on
  displacement = 0.1
  sw_vibrational_modes = on
  point_group = Td
  sw_lattice_dielectric_tensor = off
}
```

GaAs の構造の点群は T_d であるので、**point_group** にそれを指定しています。**sw_calc_force** と **sw_vibrational_modes** がともに ON になっているので、振動解析のための力計算を行い、それがすべて完了したあとに、振動解析が行われます。

SCF 計算行ったときと同じ様にして、**PHASE** を実行してください。

```
% mpirun -np 1 PATH_TO_PHASE0/bin/phase
```

この計算が終わると **force.data** が出力されます。

先の入力で **sw_calc_force** を OFF にし、**sw_lattice_dielectric_tensor** を ON にして、格子誘電率を計算します。**berry.data** を読み込み有効電荷を計算するために、下記の様に **sw_effective_charge** を ON にしなければいけません。

```
Postprocessing{
  Polarization{
    sw_bp_property = on
    property = effective_charge
  }
}
```

先ほどと同様に **PHASE** を実行します。

```
% mpirun -np 1 PATH_TO_PHASE0/bin/phase
```

この計算が終わると、出力ファイル **mode.data** に振動解析の結果、モード有効電荷、格子誘電率が出力されます。

```
n=      6 T2  IR&R
hbarW=  0.12314123E-02 Ha =  0.33508432E-01 eV; nu=  0.27026376E+03 cm^-1
  1   0.7197015024  0.0000000000  0.0000000000
  2  -0.6942836217  0.0000000000  0.0000000000
Mode effective charge and its norm:
Z=  0.3438767680  0.0000000000  0.0000000000 Norm=  0.3438767680
--- Lattice and static dielectric tensors ---
[   1.8176   0.0000   0.0000 ] [   1.8176   0.0000   0.0000 ]
[   0.0000   1.8176   0.0000 ] [   0.0000   1.8176   0.0000 ]
[   0.0000   0.0000   1.8176 ] [   0.0000   0.0000   1.8176 ]
```

振動解析だけを行ったときとは違い、各モードの振動数と固有ベクトルの次にモード有効電荷が出力されていま

す。'Z='の次に並んでいる三つの値がモード有効電荷のデカルト座標での三成分です。'Norm='の次の値はこのベクトルの大きさです。最後の三行に格子誘電率が出力されています。左側の 3x3 の行列が格子誘電率です。右側にはその格子誘電率に電子誘電率を加えた行列が出力されます。この計算では、電子誘電率を入力しなかったもので、電子誘電率の値はゼロになっています。格子誘電率は 1.8 と計算されています。ボルン有効電荷は output001 に出力されています。

--- Corrected effective charges ---				
Zeff(1) =	[2.06653	0.00000	0.00000]
	[0.00000	2.06653	0.00000]
	[0.00000	0.00000	2.06653]
Zeff(2) =	[-2.06653	0.00000	0.00000]
	[0.00000	-2.06653	0.00000]
	[0.00000	0.00000	-2.06653]

'Corrected effective charges'というタイトル以下に書かれている 3x3 の行列がボルン有効電荷テンソルです。'Zeff'の後の括弧の中の番号は原子の番号です。Ga 原子のボルン有効電荷は 2.07 と計算され、As のボルン有効電荷は 2.07 と計算されています。

11.5 UVSOR-Berry-Phonon (改良版)

11.5.1 概要

prep_zeff.pl, prep_piezo.pl などの Perl スクリプトを利用して、格子誘電率や圧電定数の計算を行う方法を説明しました。PHASE/0 には、これらの Perl スクリプトを統合し、さらに機能を充実させた新しいスクリプト、berry.pl が付属しています。ここでは、この berry.pl スクリプトの利用方法を説明します。なお、以前のバージョンからあった prep_zeff.pl, prep_piezo.pl などの Perl スクリプトも変わらず利用することも可能です。

11.5.2 使用方法

berry.pl スクリプトの利用方法を説明します。基本的な使い方（テンプレート入力データの準備の仕方など）は以前のスクリプトと同様ですが、より柔軟な制御が可能となっています。

11.5.2.1 準備

berry.pl による計算を行う前に、「テンプレート入力ファイル」を格納したディレクトリを、実行したい計算に応じて作成する必要があります。以下、ケースごとにテンプレート入力ファイルの作成方法を説明します。

（１）ボルン有効電荷に必要なベリー位相の計算を行う場合

ボルン有効電荷を計算するために必要なベリー位相は、格子誘電率および圧電定数の内部ひずみ項の計算に必要です。対称性から等価でないすべての原子を x, y, z の 3 方向に変位させ、さらに各々のケースで 3 つの逆格子ベクトルに対応するベリー位相を計算する必要があります。変位をさせないケースも必要なので、原子数 $\times 3 \times 3 + 3$ 種類のベリー位相を評価する計算を行います。

● SCF 計算のテンプレートディレクトリ

まず、通常の PHASE の SCF 計算用の入力データを用意します。原子座標は、構造最適化が施されているものを指定します。また、上述のように原子を変位させるので、対称性の指定は行わないようにします。この入力データを、任意のディレクトリ（たとえば `template_scf`）に置きます。file_names.data ファイルも通常通り置きます。この入力パラメーターファイルに、以下を加えます。

```
structure{
  atom_list{
    sw_displace_atom = on
    displaced_atom = <ATOM_ID>
    ux = <Ux>
    uy = <Uy>
    uz = <Uz>
  }
}
```

<ATOM_ID>, <Ux>, <Uy>, <Uz>は berry.pl によって実際に利用する値に置き換えられるプレースホルダーです。大文字、小文字の違いも含め、この通りに記述する必要があります。

● ベリー位相計算のテンプレートディレクトリ

つぎに、ベリー位相計算用のテンプレートディレクトリ（たとえば `template_berry`）を、SCF テンプレートディレクトリと同じ階層に作成します。このテンプレートディレクトリの下に、ekcal 用の入力データを作成します。通常の ekcal 入力に、以下を加えます。

```
structure{
  atom_list{
```

```

        sw_displace_atom = on
        displaced_atom = <ATOM_ID>
        ux = <Ux>
        uy = <Uy>
        uz = <Uz>
    }
}
...
Berry_phase{
    sw_berry_phase = on
    g_index = <G_INDEX>
    mesh{
        n1 = <MESH_N1>
        n2 = <MESH_N2>
        J = <MESH_J>
    }
}

```

<ATOM_ID>, <Ux>, <Uy>, <Uz>は SCF 計算の場合と同様です。<G_INDEX>, <MESH_N1>, <MESH_N2>, <MESH_J>は計算実行時にベリー位相計算に必要なメッシュパラメーターに置き換わるプレースホルダーです。大文字、小文字の違いも含め、この通りに記述する必要があります。

また、file_names.data は以下のように記述する必要があります。

```

&fnames
F_INP = './nfinp.data'
....
F_CHGT = './<SCF_DIR>/nfchgt.data'
/

```

<SCF_DIR>は、計算実行時に対応する SCF 計算ディレクトリ名に置き換わるプレースホルダーです。大文字、小文字の違いも含め、この通りに記述する必要があります。

※ 3D 版の場合の注意点

ベリー位相計算は、ekcal ではなく phase を用いて行ってください。この際、以下の要領で“**k**点を一点ずつ処理する”モードにしてください。

```

control{
    fixed_charge_option{
        kparallel = one_by_one
    }
}

```

(2) 圧電定数のイオン固定項の計算に必要なベリー位相を計算する場合

圧電定数のイオン固定項を計算する場合、ひずみ下でのベリー位相が必要となります。対称性から 0 にならないひずみ成分を指定し、各ひずみ成分に対して 3 つの逆格子ベクトルにそったベリー位相を計算します。必要なベリー位相の数は、参照データとしてひずんでいない系のベリー位相も計算するので、“0 にならない成分の数×3+3”です。この計算の場合も、ボルン電荷の場合と同様に SCF 計算用およびベリー位相計算用のテンプレートディレクトリーを作成します。

● SCF 計算用のテンプレートディレクトリ

ボルン電荷の場合は原子を変位させた計算を行います。圧電定数のイオン固定項の場合は単位胞をひずませた計算を行います。ボルン有効電荷の場合と同じように、SCF 計算用の入力パラメーターファイルをテンプレートディレクトリの下に置き、以下のような設定を施します。

```
structure{
  strain{
    sw_strained_cell = on
    e11 = <E11>
    e22 = <E22>
    e33 = <E33>
    e23 = <E23>
    e32 = <E32>
    e31 = <E31>
    e13 = <E13>
    e12 = <E12>
    e21 = <E21>
  }
}
```

<E11>, ... は計算実行時に設定したひずみに置き換わるプレースホルダーです。大文字、小文字の違いも含め、この通りに記述する必要があります。

● ベリー位相計算のテンプレートディレクトリ

ボルン有効電荷の場合と同じように、ekcal 用の入力パラメーターファイルをベリー位相計算用に作成したテンプレートディレクトリの下に置き、以下のように編集します。

```
structure{
  strain{
    sw_strained_cell = on
    e11 = <E11>
    e22 = <E22>
    e33 = <E33>
    e23 = <E23>
    e32 = <E32>
    e31 = <E31>
    e13 = <E13>
    e12 = <E12>
    e21 = <E21>
  }
}
..
Berry_phase{
  sw_berry_phase = on
  g_index = <G_INDEX>
  mesh{
    n1 = <MESH_N1>
    n2 = <MESH_N2>
    J = <MESH_J>
  }
}
```

<E11>, ... は SCF 計算の場合と同様です。また、<MESH_N1>, <MESH_N2>, <MESH_J> はボルン電荷の場合と同様、計算実行時にベリー位相計算のメッシュパラメーターに置き換わります。また、ボルン電荷計算の場合と同様 file_names.data は以下のように作成する必要があります。

```
&fname$
F_INP = './nfinp.data'
....
F_CHGT = '../<SCF_DIR>/nfchgt.data'
/
```

(3) 圧電定数の内部ひずみ項を計算するために必要な、ひずみ下での原子間力を計算する場合
 この場合、作成する必要があるのは **SCF** 計算用のテンプレートディレクトリのみです。テンプレートディレクトリを作成し、その下に通常の**SCF**計算用の入力パラメータファイルと **file_names.data** ファイルを置きます。**SCF** 計算用の入力パラメータファイルには以下の内容を追加します。

```
structure{
  strain{
    sw_strained_cell = on
    e11 = <E11>
    e22 = <E22>
    e33 = <E33>
    e23 = <E23>
    e32 = <E32>
    e31 = <E31>
    e13 = <E13>
    e12 = <E12>
    e21 = <E21>
  }
}
```

11.5.2.2 コントロールファイルの記述

berry.pl の振る舞いは、コントロールファイルを介して指定します。たとえば、以下のような内容になります (#で始まる文はコメント文)。

```
#overall control
property = zeff
cpumax = 1000

#directories under which the template files reside
template_scf = scf
template_berry = berry

#parameters for the berry-phase calculation
atom_list = 1 3
strain_list = 1 3 5
displacement = 0.1
strain = 0.01
mesh1 = 6 6 15
mesh2 = 6 6 15
mesh3 = 6 6 15

#execution control
np = 4
ndir = 2
```

```

ne = 1
nk = 2
ne_b = 2
scf_command = mpiexec -np NP phase ne=NE nk=NK
berry_command = mpiexec -np NP ekcal ne=NE_B

#unit cell info, optional
a_vector = 5.01 0.0 0.0
b_vector = 0.0 5.01 0.0
c_vector = 0.0 0.0 5.01

```

この例からわかるように、パラメータ 1 つにつき 1 行を利用し、“キーワード=値”という形式でパラメータを指定します。利用できるキーワードとそのデフォルト値は下記の通り。

キーワード	説明
property	どのような計算を行うかを指定します。zeff, piezo, strfrc のいずれかです。zeff を指定するとボルン有効電荷用のベリー位相計算, piezo を選ぶと圧電定数のイオン固定項の計算に必要なひずみ下でのベリー位相計算, strfrc を選ぶと圧電定数の内部ひずみ項の計算に必要なひずみ下での原子間力の計算を行うことを指定することになります。デフォルト値は zeff。
cpumax	計算の最大時間を秒の単位で指定します。ここで指定した時間よりも経過時間が長い場合、計算はすみやかに終了します。0 以下の値を指定すると、この条件では計算は終了しません。デフォルト値は 1。
stopcheck	計算停止条件を満たしているかどうかをチェックする間隔を秒の単位で指定します。デフォルト値は 10。
length_unit	コントロールファイル中に利用される長さの単位を指定します。bohr, angstrom, nm のいずれかを指定します。デフォルト値は bohr。
template_scf	SCF 計算用のテンプレートディレクトリーのディレクトリ名を指定します。デフォルト値は template_scf。
template_berry	ベリー位相計算用のテンプレートディレクトリーのディレクトリ名を指定します。デフォルト値は template_berry。
atom_list	変位させる原子の ID を、空白区切りで指定します。property=zeff の場合必須の指定です。
strain_list	ひずみ成分を空白区切りで指定します。対応は、次の通りです。 1 → 11 成分 2 → 22 成分 3 → 33 成分 4 → 23 成分 5 → 13 成分 6 → 12 成分 property=piezo および strfrc の場合必須の指定です。
displacement	原子の変位量を指定します。property = zeff の場合に指定する必要があります。デフォルト値は 0.1 bohr。
strain	ひずみの量を指定します。property = piezo および strfrc の場合に指定する必要があります。デフォルト値は 0.01。
mesh1	1 番目の逆格子ベクトルに沿ったベリー位相計算のメッシュパラメーターを空白区切りで n1 n2 J のように指定します。property = zeff および piezo の場合必須の指定です。
mesh2	2 番目の逆格子ベクトルに沿ったベリー位相計算のメッシュパラメーターを空白区切りで n1 n2 J のように指定します。property = zeff および piezo の場合必須の指定です。
mesh3	3 番目の逆格子ベクトルに沿ったベリー位相計算のメッシュパラメーターを空白区

	切りで n1 n2 J のように指定します。property = zeff および piezo の場合必須の指定です。
np	MPI プロセス数を指定します。デフォルト値は 1。
ndir	ディレクトリ並列数を指定します。デフォルト値は 1。
ne	バンド並列数を指定します。デフォルト値は 1。
nk	k 点並列数を指定します。デフォルト値は 1。
ng	(三次元版のみ) G 点並列数を指定します。デフォルト値は 1。
ne_b	ベリー位相計算時におけるバンド並列数を指定します。デフォルト値は 1。
ng_b	(三次元版のみ) ベリー位相計算時における G 並列数を指定します。デフォルト値は 1。
scf_command	SCF 計算の実行方法を指定します。たとえば, scf_command = mpirun -np NP phase ne=NE nk=NK (2D 版) scf_command = mpirun -np NP phase ne=NE nk=NK ng=NG (3D 版) などと指定します。NP, NE, NK, NG は、計算実行時に上述の np, ne, nk, ng に置き換わります。ただし、ディレクトリ並列の数によっては“あまり”が発生することがあり、その場合は ne=NE nk=NK ng=NG は省略されて計算が投入されます。デフォルト値は mpirun phase ですが、利用している環境に合わせて適切な指定をする必要があります。
berry_command	ベリー位相計算の実行方法を指定します。たとえば, berry_command = mpirun -np NP ekcal ne=NE_B (2D 版) berry_command=mpirun -np NP phase ne=NE_B ng=NG_B (3D 版) などと指定します。NE_B は上述の ne_b に、NG_B は ng_b に置き換わります。デフォルト値は mpirun ekcal です、利用している環境に合わせて適切な指定をする必要があります。
a_vector	<i>a</i> 軸の三成分を空白区切りで指定します。必須ではありませんが、指定しておくくとベリー位相計算のメッシュパラメーターの参照値を算出します。
b_vector	<i>b</i> 軸の三成分を空白区切りで指定します。必須ではありませんが、指定しておくくとベリー位相計算のメッシュパラメーターの参照値を算出します。
c_vector	<i>c</i> 軸の三成分を空白区切りで指定します。必須ではありませんが、指定しておくくとベリー位相計算のメッシュパラメーターの参照値を算出します。

※ 3D 版の場合、ベリー位相計算には phase を使うようにしてください。

11.5.2.3 並列計算について

並列計算の設定について注意すべき点を挙げます。berry.pl による計算は、通常のパンド、**k** 点による並列に加え、複数のディレクトリにまたがって並列計算を行う“ディレクトリ並列”によって行われます。したがって、ディレクトリ並列数 (パラメーター ndir) を 2 以上にする場合、SCF 計算の場合は $np=ne \times nk$ ではなく $np=ndir \times ne \times nk$ となるように、ベリー位相計算の場合は $np=ne_b$ ではなく $np=ne_b \times ndir$ となるように並列数を調整してください。また、SCF 計算とベリー位相計算とでバンド並列数が異なるのは、ベリー位相計算は **k** 点並列に未対応のためです。

11.5.2.4 スクリプトの実行

berry.pl を引数なしで実行すると、以下のようなメッセージが得られます。

```
% berry.pl
Usage : berry.pl control [OPTIONS]
```

第一引数にコントロールファイルのファイル名を指定し、さらに必要に応じてオプションを指定して制御する仕組みになっています。

以下のようなコマンドを実行すると、コントロールファイルの解釈と解析のみ行います。

```
% berry.pl control --mode=analyze
```


以下のようなコマンドを実行すると、コントロールファイルの解釈と解析のあと、計算用のディレクトリを作成します。

```
% berry.pl control --mode=gendir
```

以下のようなコマンドを実行すると、コントロールファイルの解釈と解析のあと、計算用のディレクトリを作成し、さらに計算を実行します。

```
% berry.pl control --mode=exec
```

--mode オプションのデフォルト値は gendir です。

11.5.2.5 計算のストップ/リスタート

計算は、以下の条件のいずれかが満たされれば終了します。

- すべての計算が終了した。
- コントロールファイルの **cpumax** で指定した時間を経過時間が超えた。
- 作業ディレクトリに **stop** という名前のファイルが作成された。

一方、リスタートは特に設定を行わずとも実行されます。計算を最初からやり直したい場合は、まず以下のコマンドによって計算用ディレクトリを削除します。

```
% berry.pl --clean
```

リスタートは、通常の状態の場合はディレクトリ単位でのリスタートとなります。すなわち、あるディレクトリにおいて途中で停止された計算は、そのディレクトリにおいて最初から行われます。各ディレクトリにおける計算をも継続して行う場合は、テンプレート入力の入力パラメータファイルに以下の設定を記述する必要があります。

SCF 計算の場合：

```
control{  
    condition = automatic  
}
```

ベリー位相計算の場合：

```
control{  
    condition = fixed_charge_automatic  
}
```

このような設定を施しておくことによって、計算が継続可能かどうかを PHASE が判断し、可能な場合は継続計算を実行するようになります。

11.5.2.6 注意点

(1) ベリー位相計算用のメッシュ

ベリー位相の計算においては、対象としたい逆格子ベクトル \mathbf{b}_i に垂直な面の面積分と \mathbf{b}_i に沿った線積分が実行されます。面積分のメッシュを n1 n2 で、線積分のメッシュを J で指定します。 \mathbf{b}_i 以外の 2 つの逆格子ベクトルを \mathbf{b}_j とすると \mathbf{b}_j を \mathbf{b}_i に垂直な面に射影したベクトル、 $\mathbf{b}_j - |\mathbf{b}_j| \cos \theta_{ji} \frac{\mathbf{b}_i}{|\mathbf{b}_i|}$, が面積分のメッシュの見積もり

の基準となるのでその長さから決めます。線積分のパラメータは、 \mathbf{b}_i の長さをもとに決定します。コントロールファイルに `a_vector`, `b_vector`, `c_vector` の指定を行っておくと、この長さの計算 (bohr⁻¹ 単位) とそこから見積もられる参考のメッシュパラメーターが以下のように標準出力に出力されます (あくまで参照値であり、得られる結果の妥当性を保証するものではありません)。

```
|b_para1|, |b_para2| and |b_perp| (in bohr^-1 units)
for reciprocal vector no. 1 : 0.172224346323159, 0.107572987734313, 0.198867545420854
for reciprocal vector no. 2 : 0.172224346322494, 0.107572987734313, 0.198867545421622
for reciprocal vector no. 3 : 0.198867545420854, 0.198867545421622, 0.107572987734313

reference value for mesh parameters n1, n2 and J
for reciprocal vector no. 1 : 8, 5, 19
for reciprocal vector no. 2 : 8, 5, 19
for reciprocal vector no. 3 : 9, 9, 10
```

(2) ボルン電荷を計算するときに指定する原子
ボルン電荷の計算を行う際に指定する原子は、対称性から等価でない原子のみを指定します (等価な原子を指定しても結果は正しくですが、不要な計算を実行することになります)。たとえば、SiO₂ の場合 Si と O をどれか 1 つずつ選びます。

(3) 並列数の指定
並列の指定は、以下が満たされるように行ってください。

```
SCF 計算の場合 : np = ndir × ne × nk
ベリー位相計算の場合 : np = ndir × ne_b
```

ここで `ndir` は並列に扱いたいディレクトリの数、`ne`, `nk` はそれぞれ SCF 計算のバンド並列および \mathbf{k} 点並列数、`ne_b` はベリー位相計算時のバンド並列数です。SCF 計算とベリー位相計算でバンド並列数が異なるのは、ベリー位相計算は \mathbf{k} 点並列に未対応のためです。この関係が成立しない場合、`berry.pl` はその旨を出力し終了します。

(4) 3 次元版について
UVSOR-Berry-Phonon は、3 次元版は対応していません。

11.5.3 計算例題 : AlN の格子誘電率, 圧電定数

例題として、AlN を取り上げます。格子誘電率, 圧電定数のイオン固定項, 圧電定数の内部ひずみ項の計算例を紹介します。各計算は、成すべき手続きが多いので、1 ステップずつ解説していきます。なお、ここで利用するサンプルデータは `samples/uvsor/lattice/AlN` 以下にあります。以降の説明は、このディレクトリにいるものとして行います。

11.5.3.1 AlN の格子誘電率

(1) 格子振動解析
まずは、格子振動解析を行います。`phonon` の下に格子振動解析を行うための入力データがおかれています。この計算は、通常の PHASE の計算と同じように実行します。

(2) ベリー位相の計算
ついで、ベリー位相の計算を行います。利用する入力テンプレートディレクトリは、`berry` 以下にあります。

```
% cd berry
% ls
```

```
control template_berry template_scf
```

control が berry.pl のコントロールファイルです。template_berry, template_scf はそれぞれベリー位相および SCF 計算用の入力テンプレートディレクトリです。その内容をみると、入力パラメータファイルには <ATOM_ID>, <Ux> など、通常の PHASE の計算では利用しない文字列が確認できます。これらの文字列は、berry.pl 実行時に適切な値に置き換わる仕組みになっています。

control ファイルは、以下のような内容です。

```
property=zeff
cpumax=10000

template_scf = template_scf
template_berry = template_berry

atom_list = 1 3
displacement = 0.1
mesh1 = 6 6 15
mesh2 = 6 6 15
mesh3 = 6 6 15

np = 8
ndir = 2
ne = 2
nk = 2
ne_b = 4
scf_command = mpiexec -n NP $HOME/phase0_2015.01/bin/phase ne=NE nk=NK
berry_command = mpiexec -n NP $HOME/phase0_2015.01/bin/ekcal ne=NE_B
```

このうち、変更する必要があるのは実行制御部分です。np, ndir, ne, nk, ne_b を利用したい MPI プロセス数に合わせて編集します。また、scf_command に PHASE の実行方法を、berry_command に ekcal の実行方法を指定します。コントロールファイルの実行制御部分を適切に編集したら、以下の要領で berry.pl を実行します。

```
% berry.pl control --mode=exec
```

この例では、1 番目と 3 番目の原子 (Al と N) を変位させてベリー位相の計算を行うので、合計で 7 の SCF 計算と 21 のベリー位相の計算が行われます。

計算がすべて終了すると、作業ディレクトリに berry.data というファイルが作成されますが、これを格子振動解析を行ったディレクトリ (この例では phonon) にコピーします。

```
% cp berry.data ../phonon
```

(3) 格子誘電率の計算

格子振動解析とベリー位相の計算が終了すれば、格子誘電率を計算するための準備はできています。まず、格子振動解析のディレクトリへ移ります。

```
% cd ../phonon
```

格子振動解析の入力パラメーターファイルに、以下の変更を施します。

- phonon ブロックの sw_calc_force を off とする。
- sw_lattice_dielectric_tensor を on とする。
- postprocessing ブロックの下で polarization ブロックの下で、sw_bp_property を on とし、property を effective_charge とする。

以下に、具体的な変更箇所を示しました。赤字で示した部分が変更を要する部分です。

```

Phonon{
  sw_phonon = on
  sw_vibrational_modes = on
  sw_calc_force = off
  sw_lattice_dielectric_tensor = on
}
postprocessing{
  polarization{
    sw_bp_property = on
    property = effective_charge
  }
}

```

この状態で **PHASE** を実行します。この計算の負荷は非常に軽いので、通常並列で実行する必要はありません。

計算結果は、**outputxxx** ファイルと **mode.data** ファイルに記録されます。**outputxxx** ファイルには、ボルン有効電荷の計算結果が記録されます。最終結果は、**--- Corrected effective charges ---**のあとに記録されます。今の例の場合、以下のようにすれば計算されたボルン有効電荷を抽出することができます（格子誘電率計算のログファイルを **output001** とします）。

```

% grep -A16 'Corrected effective' output001
--- Corrected effective charges ---
      [ 2.50954 0.00000 0.00000 ]
Zeff( 1) = [ 0.00000 2.50954 0.00000 ]
      [ 0.00000 0.00000 2.64146 ]

      [ 2.50954 0.00000 0.00000 ]
Zeff( 2) = [ 0.00000 2.50954 0.00000 ]
      [ 0.00000 0.00000 2.64146 ]

      [ -2.50954 0.00000 0.00000 ]
Zeff( 3) = [ 0.00000 -2.50954 0.00000 ]
      [ 0.00000 0.00000 -2.64146 ]

      [ -2.50954 0.00000 0.00000 ]
Zeff( 4) = [ 0.00000 -2.50954 0.00000 ]
      [ 0.00000 0.00000 -2.64146 ]

```

格子誘電率は、**mode.data** ファイルの最後に、以下のような形式で記録されます。

```

--- Lattice and static dielectric tensors ---
[ 3.7058 0.0000 0.0000 ] [ 3.7058 0.0000 0.0000 ]
[ 0.0000 3.7058 0.0000 ] [ 0.0000 3.7058 0.0000 ]
[ 0.0000 0.0000 4.9314 ] [ 0.0000 0.0000 4.9314 ]

```

結果が 2 組記録されます。1 つめが格子誘電率の計算結果ですが、2 つめは電子系誘電率の記述が入力パラメータファイルにあった場合それを加えた結果が出力されます。記述がない場合、1 つめと同じ結果が出力されます。

11.5.3.2 AlN の圧電定数のイオン固定項

(1) ベリー位相の計算

圧電定数のイオン固定項を計算するためには、ひずみ下でのベリー位相が必要です。そのような計算を行うため

の入力テンプレートディレクトリが `piezo/clamped` 以下にあります。

```
% cd piezo/clamped
% ls
control template_berry template_scf
```

`control` が `berry.pl` 用のコントロールファイル、`template_berry` と `template_scf` がそれぞれベリー位相計算および SCF 計算用の入力テンプレートディレクトリです。その内容をみると、入力パラメータファイルには `<E11>`、`<E22>` など、通常の PHASE の計算では利用しない文字列が確認できます。これらの文字列は、`berry.pl` 実行時に適切な値に置き換わる仕組みになっています。

`control` ファイルの内容は 11.5.3.1 の場合とほぼ同じですが、以下に示すように `strain_list` と `strain` の設定がなされている点が異なります。また、`atom_list` と `displacement` は不要なので消してあります（指定があっても問題はありません）。

```
property=zeff
cpumax=10000
...
...
strain_list = 1 3 5
strain = 0.01
mesh1 = 6 6 15
mesh2 = 6 6 15
mesh3 = 6 6 15
...
...
```

実行制御部分の変更は、11.5.3.1 の場合と同じように行います。

コントロールファイルの実行制御部分を適切に編集したら、以下の要領で `berry.pl` を実行します。

```
% berry.pl control --mode=exec
```

この例では、1, 3, 5 成分のひずみを与えてベリー位相の計算を行うので、ひずみのない場合もいれて合計で 4 の SCF 計算と 12 のベリー位相の計算が行われます。

計算がすべて終了すると、作業ディレクトリに `berry.data` というファイルが作成されますが、これを `scf_e0` ディレクトリ（`berry.pl` によって作成されたディレクトリの 1 つ）にコピーします。

```
% cp berry.data scf_e0/
```

（2）圧電定数のイオン固定項の計算

圧電定数のイオン固定項は、`scf_e0` ディレクトリにおいて行います。

```
% cd scf_e0
```

このディレクトリにある入力パラメータファイルに、以下の変更を施します。

- `condition` を `continuation` とする。
- `postprocessing` ブロックの下の `polarization` ブロックにおいて、変数 `sw_bp_property` を `on` とし、`property` に `piezoelectric_const` を指定する。

以下に、具体的な変更箇所を示します。赤字で示した部分が変更を要する部分です。

```
Control{
    condition = continuation
}
...
...
postprocessing{
    polarization{
```

```

    sw_bp_property = on
    property = piezoelectric_const
}
}

```

この状態で **PHASE** を実行します。この計算の負荷は非常に軽いので、通常並列で実行する必要はありません。結果は、**outputxxx** ファイルに、Piezoelectric constant のあとに原子単位および C/m^2 単位で記録されます。以下の要領でこの情報を抽出することができます。

```

% grep -A6 'Piezoelectric' output001
=== Piezoelectric constant (a.u.) ===
  1    -0.0000008823    0.0000000060    0.0043842675
  2     0.0000000000    0.0000000000    0.0000000000
  3    -0.0000000034    0.0000000049   -0.0069512745
  4     0.0000000000    0.0000000000    0.0000000000
  5     0.0058046970   -0.0000000097    0.0000525809
  6     0.0000000000    0.0000000000    0.0000000000
=== Piezoelectric constant (C/m^2) ===
  1    -0.0000504820    0.0000003433    0.2508448251
  2     0.0000000000    0.0000000000    0.0000000000
  3    -0.0000001971    0.0000002824   -0.3977155187
  4     0.0000000000    0.0000000000    0.0000000000
  5     0.3321143610   -0.0000005530    0.0030084060
  6     0.0000000000    0.0000000000    0.0000000000

```

0 になるべき項が 0 になっていないのは、数値誤差によるものです。

11.5.3.3 AlN の圧電定数の内部ひずみ項

(1) ひずみ下における原子間力の計算

圧電定数の内部ひずみ項を計算するためには、ひずみ下における原子間力が必要です。この計算を行うための例題が **piezo/internal** 以下にあります。

```

% cd piezo/internal
% ls
control template_scf

```

このケースではベリー位相の計算は不要なので、テンプレート入力ディレクトリは **SCF** 計算用のもののみとなっています。

template_scf の内容は、11.5.3.2 の同名のディレクトリと全く同じ内容です。ひずみを与える計算を行うため、入力パラメータファイルにプレースホルダーが記述されています。**control** ファイルの内容もほぼ同じですが、**property** 変数には **strfrc** が指定されています。また、実行制御部分を環境に合わせて書き換える必要がある点もこれまでと同様です。

実行制御部分を書き換えたら、**berry.pl** を実行します。

```

% berry.pl control --mode=exec

```

3 つのひずみ成分に対して、正と負のひずみを与えた計算を実行するので、計 6 つの **SCF** 計算を実行することになります。

計算が終了したら、**strfrc.data** というファイルが作成されます。このデータと格子振動解析の結果を利用して内部ひずみ項を計算します。格子振動解析は、11.5.3.1 で実行した結果を再度利用するので、この格子振動解析を行ったディレクトリに **strfrc.data** ファイルをコピーします。

```

% cp strfrc.data ../../phonon

```

(2) 圧電定数の内部ひずみ項の計算

圧電定数の内部ひずみ項は、上述の **phonon** ディレクトリにおいて行います。

```
% cd ../../phonon
```

内部ひずみ項の計算にはここで計算した **strfric.data** のほかに原子を変位させたときのベリ一位相も必要であるが、ベリ一位相の計算は 11.5.3.1 において実行済みなのでこれを再利用します。

phonon ディレクトリの入力パラメータファイルに、以下の変更を施します (大体の変更は、11.5.3.1 において施したもの)。

- **phonon** ブロックの **sw_calc_force** を **off** とする。
- **sw_internal_strain_piezoelectric_tensor** を **on** とする。
- **sw_lattice_dielectric_tensor** を **off** とする。
- **postprocessing** ブロックの下に **polarization** ブロックの下で、**sw_bp_property** を **on** とし、**property** を **effective_charge** とする。

以下に、具体的な変更箇所を示します。赤字で示した部分の変更を要する部分です。

```
Phonon{
  sw_phonon = on
  sw_vibrational_modes = on
  sw_calc_force = off
  sw_internal_strain_piezoelectric_tensor = on
  sw_lattice_dielectric_tensor = off
}
postprocessing{
  polarization{
    sw_bp_property = on
    property = effective_charge
  }
}
```

この状態で **PHASE** を実行します。この計算の負荷は非常に軽いので、通常並列で実行する必要はありません。

結果は、**outputxxx** ファイルに、Internal-strain piezoelectric tensor のあとに原子単位および C/m^2 単位で記録されます。以下の要領でこの情報を抽出することができます。

```
% grep -A6 'Internal-strain' outputxxx
=== Internal-strain piezoelectric tensor (a.u.) ===
  1      0.0000000000      0.0000000000     -0.0157392802
  2      0.0000000000      0.0000000000      0.0000000000
  3      0.0000000000      0.0000000000      0.0348231759
  4      0.0000000000      0.0000000000      0.0000000000
  5     -0.0118073889      0.0000000000      0.0000000000
  6      0.0000000000      0.0000000000      0.0000000000
=== Internal-strain piezoelectric tensor (C/m^2) ===
  1      0.0000000000      0.0000000000     -0.9005191764
  2      0.0000000000      0.0000000000      0.0000000000
  3      0.0000000000      0.0000000000      1.9923997401
  4      0.0000000000      0.0000000000      0.0000000000
  5     -0.6755569503      0.0000000000      0.0000000000
  6      0.0000000000      0.0000000000      0.0000000000
```

最後に、表 11.3 に圧電定数の計算結果をまとめます。

表 11.3 AlN の圧電定数 ; 単位は C/m^2

成分	イオン固定項	内部ひずみ項	合計	実験値
----	--------	--------	----	-----

e_{31}	0.251	-0.901	-0.650	-0.58
e_{33}	-0.398	1.992	1.594	1.55
e_{15}	0.332	-0.676	-0.344	-0.48

11.6 線形応答時間依存密度汎関数法 (LR-TDDFT)

11.6.1 機能の概要

11.6.1.1 はじめに

独立粒子近似における励起スペクトル計算では、Kohn-Sham 方程式の解として得られた固有エネルギー準位間の遷移の計算を行います。実際に観測される励起スペクトルは、粒子間の相互作用が働くために、遷移エネルギーやピーク強度が、独立粒子近似における予測とは異なる場合があります。ここでは、その相互作用を線形応答の範囲で取り入れた Linear-response Time-dependent density functional theory (LR-TDDFT) について説明します。

11.6.1.2 固体への適用

独立粒子近似における、外場の変化に対する系の応答関数 χ^0 は

$$\chi_{GG'}^0(q, \omega) = 2 \int_{\text{BZ}} \frac{d\mathbf{k}}{(2\pi)^3} \sum_{n, n'} (f_{n\mathbf{k}-\mathbf{q}} - f_{n'\mathbf{k}}) \frac{\rho_{n'\mathbf{k}}^*(\mathbf{q}, \mathbf{G}) \rho_{n\mathbf{k}}(\mathbf{q}, \mathbf{G}')}{\omega - (\varepsilon_{n'\mathbf{k}} - \varepsilon_{n\mathbf{k}-\mathbf{q}}) + i\eta}$$

で表されます。ここで、

$$\rho_{n'\mathbf{k}}(\mathbf{q}, \mathbf{G}) = \langle n' \mathbf{k} | e^{i(\mathbf{q}+\mathbf{G})\cdot\mathbf{r}} | n\mathbf{k} - \mathbf{q} \rangle$$

です。一方、粒子間のクーロン及び交換相互作用を取り入れた場合の応答関数 χ は、 χ^0 と Dyson 方程式

$$\chi = \chi^0 + \chi^0(v + f_{\text{xc}})\chi$$

の関係にあります。ここで、 v はクーロンカーネルで、

$$v_{\mathbf{G}}(\mathbf{q}) = \frac{4\pi}{|\mathbf{q} + \mathbf{G}|^2}$$

で表されます。一方、 f_{xc} は交換相関カーネルですが、具体的な表式は定まっておらずいくつかのモデルが提案されています。以下に採用したモデルを記します。

- RPA (Random Phase Approximation)

$$f_{\text{xc}} = 0$$

- LRC (Long range correction)

$$f_{\text{xc}} = -\frac{\alpha}{|\mathbf{q} + \mathbf{G}|^2}$$

さて、実験と比較可能な量としては巨視的な誘電関数 ε_M があります。これは、

$$\varepsilon_M(\omega) = 1 - v_0 \bar{\chi}_{\mathbf{G}=\mathbf{G}'}=0(\omega)$$

を用いて算出します。なお、 $\bar{\chi}$ は χ に似た応答関数で、以下のようにクーロンカーネルの $\mathbf{G} = 0$ 成分を除去した関数です。

$$\begin{aligned} \bar{\chi} &= \chi_0 + \chi_0(\bar{v} + f_{\text{xc}})\bar{\chi} \\ \bar{v}(\mathbf{q}) &= \begin{cases} v_{\mathbf{G}}(\mathbf{q}) & \mathbf{G} \neq 0 \quad (134) \\ 0 & \mathbf{G} = 0 \quad (135) \end{cases} \end{aligned}$$

11.6.1.3 孤立系への適用

分子など孤立系では、 f_{xc} として以下のようなモデルを採用しました。

- ALDA (Adiabatic Local Density Approximation)

$$f_{xc}(\mathbf{r}, \mathbf{r}') = \delta(\mathbf{r} - \mathbf{r}') \frac{\partial v_{xc}(\rho(\mathbf{r}))}{\partial \rho}$$

この f_{xc} を実空間メッシュ上で評価し、フーリエ変換して得られた $f_{xc}(\mathbf{G}, \mathbf{G}')$ を式(133)に代入すれば、誘電関数等のスペクトルが計算できます。しかし、このフーリエ変換の際に、電荷密度が小さい領域で精度が悪くなることが知られています。そこで、本プログラムでは、以下の様な手法を用いています。まず、相互作用のない電子空孔グリーン関数 L^0 を

$$\lim_{q \rightarrow 0} \chi_{GG'}^0(q, \omega) = -i \sum_{nn'} \sum_{\mathbf{k}} \lim_{q \rightarrow 0} [\rho_{n' n \mathbf{k}}^*(\mathbf{q}, \mathbf{G}) \rho_{n' n \mathbf{k}}(\mathbf{q}, \mathbf{G}')] L_{nn' \mathbf{k}}^0(\omega)$$

により定義します。同様に、相互作用を含んだ電子空孔グリーン関数 \bar{L} を

$$\lim_{q \rightarrow 0} \bar{\chi}_{GG'}(q, \omega) = -i \sum_{nn'} \sum_{mm'} \sum_{\mathbf{k}, \mathbf{k}'} \lim_{q \rightarrow 0} [\rho_{n' n \mathbf{k}}^*(\mathbf{q}, \mathbf{G}) \rho_{m' m \mathbf{k}'}(\mathbf{q}, \mathbf{G}')] \bar{L}_{nn' \mathbf{k}, mm' \mathbf{k}'}(\omega)$$

で定義します。 \bar{L} と L^0 の関係は、Bethe-Salpeter 方程式

$$\bar{L}_{nn' \mathbf{k}, m' m \mathbf{k}'}(\omega) = L_{nn' \mathbf{k}}^0(\omega) \left[\delta_{nm} \delta_{n' m'} \delta_{\mathbf{k} \mathbf{k}'} + i \sum_{ss'} \sum_{\mathbf{k}_1} \Xi_{nn' \mathbf{k} ss' \mathbf{k}_1} \bar{L}_{ss' \mathbf{k}_1, mm' \mathbf{k}'}(\omega) \right]$$

で表されます。ここで、

$$\Xi_{nn' \mathbf{k} ss' \mathbf{k}_1} = -V_{nn' \mathbf{k} ss' \mathbf{k}_1} - K_{nn' \mathbf{k} ss' \mathbf{k}_1}$$

$$V_{nn' \mathbf{k} ss' \mathbf{k}_1} = \frac{1}{\Omega N_{\mathbf{k}}} \sum_{\mathbf{G} \neq 0} \rho_{nn' \mathbf{k}}(\mathbf{q} = 0, \mathbf{G}) \rho_{ss' \mathbf{k}_1}^*(\mathbf{q} = 0, \mathbf{G}) v(\mathbf{G})$$

$$K_{nn' \mathbf{k} ss' \mathbf{k}_1} = 2 \int \int d\mathbf{r} d\mathbf{r}' \phi_{n\mathbf{k}}^*(\mathbf{r}) \phi_{n'\mathbf{k}}(\mathbf{r}) f_{xc}(\mathbf{r}, \mathbf{r}') \phi_{s'\mathbf{k}'}(\mathbf{r}') \phi_{s\mathbf{k}'}^*(\mathbf{r}')$$

です。なお、 Ω は系の体積、 $N_{\mathbf{k}}$ は \mathbf{k} 点サンプリングの数です。

さて、 f_{xc} としてALDAを用いることから、実際の計算では式(143)の積分は実空間メッシュ上での1重積分になります。また、 \mathbf{k} 点として Γ 点のみを使用することから、式(138)-(140)の \mathbf{k} 点に関する和は不要となります。

さて、孤立系でしばしば計測される光吸収断面積 (Photo Adsorption Cross Section, PACS) は、

$$\sigma(\omega) = \frac{\Omega}{c} \omega \text{Im}[\epsilon_M(\omega)]$$

表されます。ここで、誘電関数 ϵ_M は、式(140)により得られた \bar{L} を式(132)及び(139)に代入することにより得ることが出来ます。

11.6.2 入力ファイルの記述

LR-TDDFT 法解析プログラムを利用するためには、次の設定が必要です。

11.6.2.1 control ブロック

まず、`phase` を用いて事前に SCF 計算を行い、系の電荷密度を求めておきます。LR-TDDFT はその電荷密度をもとに計算を行います。このため、`control` ブロック内で `condition = fixed_charge` として下さい。また、UVSOR と同様に、局所ポテンシャルが軌道ポテンシャルである TM 型擬ポテンシャルを用いる場合、`use_additional_projector = on` として下さい。

```
control{
  condition = fixed_charge
  cpumax = 1 day
}
```

```

max_iteration = 600
use_additional_projector=on
}

```

11.6.2.2 accuracy ブロック

accuracy ブロック内では、UVSOR と同様に、固有値計算のためのパラメータ設定を行います。

```

accuracy{
...
  ek_convergence{
    num_extra_bands = 0
    num_max_iteration = 2000
    sw_eval_eig_diff = on
    delta_eigenvalue = 1.e-6 rydberg
    succession      = 3
  }
...
}

```

11.6.2.3 structure ブロック

k 点に関する和を、ブリルアンゾーン内の、既約化されていない全ての k 点を用いて行うため、E 以外の対称性をオフにします。

```

structure{
...
  symmetry{
    method = manual
    tspace{
      lattice_system = primitive
      generators{
        !#tag rotation tx  ty  tz
                E      0   0   0
      }
    }
  }
...
}

```

11.6.2.4 spectrum ブロック

LR-TDDFT によるスペクトル計算に関するパラメータ設定を行います。以下、ブロック内で使用可能な変数について説明します。[]内の値はデフォルト値で、無指定の場合に使用されます。

type 変数 [OPTICS] OPTICS、PACS が使用可能です。OPTICS は誘電関数計算を行い、主として固体に使用します。PACS は光吸収断面積で、分子など孤立系に使用します。

momentum_transfer ブロック momentum transfer ベクトルに関する記述を行うブロック

deltaq 変数 [1.0E-3] momentum transfer ベクトル q の大きさを指定します。単位は \AA^{-1} です。

nx, ny, nz 変数 [0.0, 0.0, 1.0] momentum transfer ベクトル q の方向を指定します。

LongWaveApprox 変数 [ON] ON、OFF が使用可能。長波長近似 ($q \rightarrow 0$) を使用する場合には、ON を指定します。

tddft ブロック sw_tddft 変数 [OFF]	tddft に関するパラメータを設定するブロック ON、OFF が使用可能。LR-TDDFT 機能を使用する場合には ON を指定します。
solver ブロック equation 変数 [DYSON]	使用するソルバーを設定するブロック DYSON、BS が使用可能。DYSON 様方程式を使用する場合には DYSON を、Bethe-Salpeter 様方程式を使用する場合には BS を指定します。後者は、分子など孤立系の場合に使用します。
XC_Kernel ブロック kernel_type 変数 [RPA]	exchange-correlation kernel のタイプ及び関連変数を設定します。 RPA、LRC、ALDA-R が指定可能です。RPA は、exchange-correlation kernel を考慮しないモデルです。LRC は、固体などの周期系で、長距離相互作用の補正を取り入れたい場合に使用します。ALDA-R は、分子など孤立系の場合に使用します。
LRC_alpha 変数 [1.0]	kernel 変数に LRC を使用した場合に設定します。
Coulomb_Kernel ブロック sw_NLF 変数 [OFF]	coulomb kernel に関する変数を設定します。 ON、OFF が使用可能。Local field ($ \mathbf{G} > 0$) を無視する近似を行う場合には、ON を指定します。
Expansion ブロック NumGVec 変数 [100]	展開に使用する G ベクトルに関する設定を行うブロック 波数ベクトルの数を設定します。
energy ブロック low, high, step 変数	スペクトルを計算するエネルギー範囲を指定するブロック low、high にはエネルギーの最小及び最大値、step にはエネルギーの間隔を入力します。
BZ_Integration ブロック width 変数 [1.0E-4 hartree]	ブリルアンゾーン内の積分に関する設定を行うブロック Lorentzian の幅を指定します。
band_gap_correction ブロック scissor_operator 変数 [0.0]	バンドギャップ補正を顕わに指定する場合に設定します。詳細は UVSOR に準じます。 ギャップの増分値を指定します。

以下は記入例です。

```
spectrum{
  type = optics
  momentum_transfer{
    deltaq = 1.0E-3
    nx = 1.1, ny = 1.2, nz = 0.9
    LongWaveApprox = ON
  }
  tddft{
    sw_tddft = ON
    solver{
      equation = DYSON
    }
    XC_Kernel{
      kernel_type = LRC
      LRC_alpha = 0.2
    }
    Coulomb_Kernel{
      sw_NLF = OFF
    }
  }
}
```

```

    Expansion{
        NumGVec = 80
    }
}
energy{
    low = 0.0 eV
    high = 10.0 eV
    step = 0.05 eV
}
BZ_integration{
    width = 0.15 eV
}
band_gap_correction{
    scissor_operator = 0.6d0 eV
}
}

```

11.6.3 計算の実行方法

phase を用いて事前に SCF 計算を行い、系の電荷密度を計算します。

```
mpirun -np NP phase
```

LR-TDDFT の計算を行うには、以下のコマンドを使用します。

```
mpirun -np NP tdlrmain
```

ここで、NP は MPI プロセス数です。

11.6.4 出力ファイル

スペクトルデータは、`spectrum.data` に出力されます。以下のような書式をとります。

A. type 変数で OPTICS を指定した場合

#	Optical spectrum				
#		NonInteracting		Interacting	
#	Energy[eV]	Real	Imaginary	Real	Imaginary
	0.000000	8.626260	0.252860	9.678273	0.327540
	0.050000	8.627214	0.252961	9.679507	0.327682
				

第 1 カラムはエネルギー値です。第 2、3 カラムは独立粒子近似における誘電関数を出力しています。なお、Real 及び Imaginary はそれぞれ、実部および虚部に対応します。また、第 4、5 カラムは Coulomb 及び exchange-correlation kernel を取り入れた誘電関数です。

B. type 変数で OPTICS を指定した場合

#	Photo Absorption Cross Section		
#	Energy[eV]	NonInteracting	Interacting
	0.000000	0.000000	0.000000
	0.050000	0.000034	0.000012
		

第 1 カラムはエネルギー値です。第 2、3 カラムは、それぞれ独立粒子近似及び相互作用を取り入れた光吸収断面積です。

11.6.5 例題

11.6.5.1 Si 結晶の誘電スペクトル

Si 結晶の誘電スペクトルの計算例題です。計算例題は、`sample/lr-tddft/SiBulk` です。まず、`scf` ディレクトリにて電荷密度の計算を行い、次に `LRC` ディレクトリにてスペクトル計算を行います。ここでは、`exchange-correlation kernel` として `LRC` を採用しています。

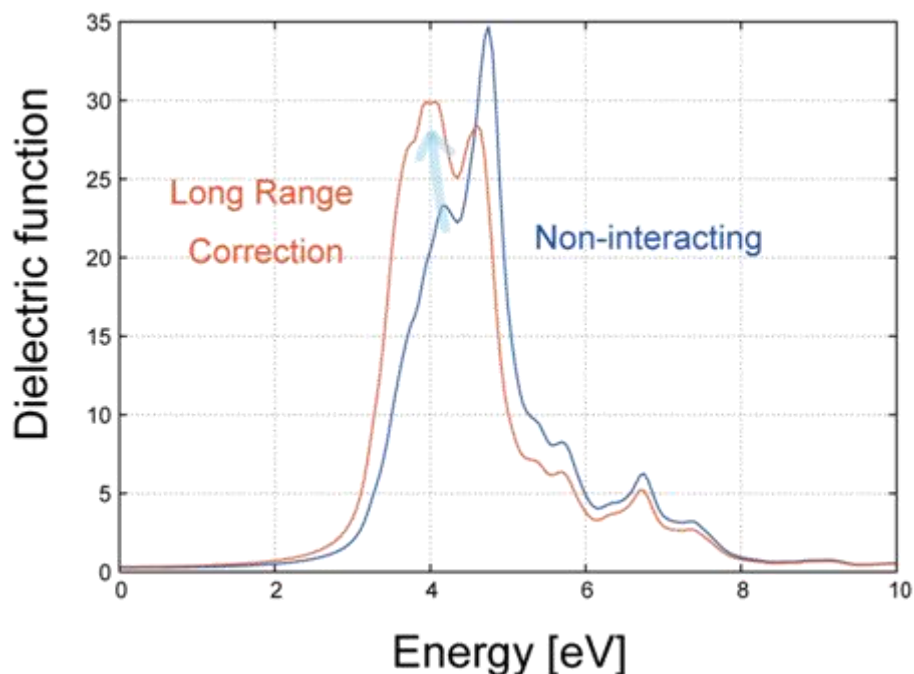


図 11-6 LRC による Si 結晶誘電スペクトルの変化。青線は独立粒子近似による結果。

図 11-6 の青線および赤線は、`spectrum.data` の虚部を表示したもので、それぞれ独立粒子近似および `LRC` を用いた場合の誘電スペクトルです。長距離相互作用の補正を行うことにより、第 1 ピークの強度が強くなる様子が分かります。なお、本系では、`TDDFT` によってもギャップ値の改善は見られません。これは、結晶のように波動関数が広がっている場合には、その間のクーロン相互作用が弱いからです。

11.6.5.2 C₆H₆ 分子の光吸収断面積

C₆H₆ 分子の光吸収断面積計算の計算例題です。計算例題は、`sample/lr-tddft/C6H6` です。まず、`scf` ディレクトリにて電荷密度の計算を行い、次に、`ALDA` ディレクトリにて移動してスペクトル計算を行います。`exchange-correlation kernel` として `ALDA` を採用しています。

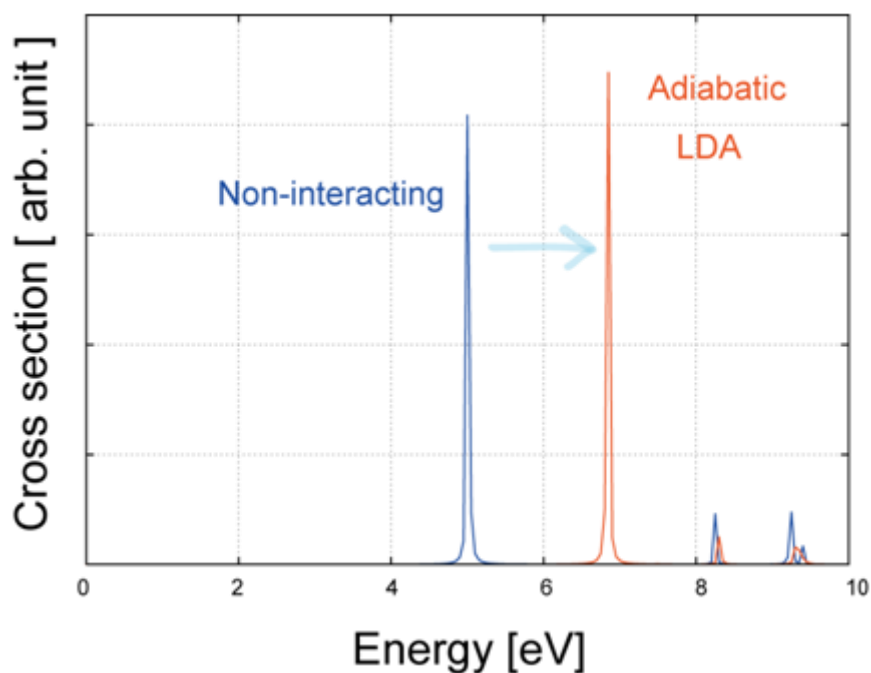


図 11-7 ALDA による C_6H_6 分子の光吸収断面積スペクトルの変化。青線は独立粒子近似による結果。

図 11-7 の青線および赤線は、`spectrum.data` を表示したもので、それぞれ独立粒子近似および ALDA を用いた場合の誘電スペクトルです。第 1 ピークのエネルギー位置が高エネルギー側にシフト、すなわちギャップ値が拡大している様子が分かります。

11.6.6 使用上の注意点

- 対称性を用いた k 点の縮約に対応していません。このため、`symmetry` ブロック中に“E”のみの対称性を指定してください。
- `solver` で `equation =BS` を指定した場合には、非磁性 (paramagnetic) な系のみを取り扱うことができます。

11.7 付録

11.7.1 eps_file

機能：file_names.data の F_EPSOUT で指定されるファイルに出力される誘電関数計算結果（テンソル形式 χ ）を読み込み、実部と虚部に分離してそれぞれ別のファイルに書き出す。

入力：誘電関数結果ファイル；名称は eps.data

出力：誘電関数実部ファイル；r_eps.data

〃 虚部ファイル；i_eps.data

使用法：ソースコード eps_file.f90 を適当な Fortran コンパイラでコンパイルし、eps.data を含むディレクトリで実行する。

11.7.2 nlo_file

機能：file_names.data の F_NLO で指定されるファイルに出力される非線形光学感受率計算結果を読み込み、必要なテンソル成分の結果を別のファイルに書き出す。

入力：感受率計算結果ファイル(名称は nlo.data)

テンソル成分

出力：入力で指定された成分の感受率計算結果ファイル；名称は nlo_abc.data。

(abc)は入力で指定されるテンソル成分。

使用法：ソースコード nlo_file.f90 を適当な Fortran コンパイラでコンパイルし、nlo.data を含むディレクトリで実行する。実行時に、成分をキーボード入力する。

使用例 1：実行体の名称が nlofile であり、zzz 成分の計算結果を別のファイルに書き出す場合。

```
% nlofile (1)
zzz (2)
line number of nlo_zzz.data = 1000 (3)
```

(1) プログラムの実行

(2) キーボード入力待ちとなるので、テンソル成分を小文字で入力する

(3) nlo_zzz.data に 1000 行のデータが出力された（メッセージ）

使用例 2：実行体の名称が nlofile であり、xxyy 成分の計算結果を別のファイルに書き出す場合。

```
% nlofile (1)
xxyy (2)
line number of nlo_xxyy.data = 1000 (3)
```

(1) プログラムの実行

(2) キーボード入力待ちとなるので、テンソル成分を小文字で入力する

(3) nlo_xxyy.data に 1000 行のデータが出力された（メッセージ）

11.8 参考文献

- [1] Gunther Harbeke, "Optical Properties of Semiconductors" in Optical Properties of Solids; F. Abeles(Ed.), North-Holland, Amsterdam (1972):Chapter 2.
- [2] A. F. Starace, "Length and Velocity Formulas in Approximate Oscillator-Strength Calculations", Phys. Rev. A, pp. 1242-1245 (1972).
- [3] A. J. Read and Needs, "Calculation of Optical Matrix Elements with Nonlocal Pseudopotentials", Phys. Rev. B, pp13071-13073 (1991).
- [4] H. Kageshima and K. Shiraishi, "Momentum-matrix-element Calculation using Pseudopotentials", Phys. Rev. B, pp14985-14992 (1997).
- [5] G. Lehmann and M. Taut, "On the Numerical Calculation of the Density of States and Related Properties", phys. stat. sol. (b), pp469-477 (1972).
- [11] B. Adolf, K. Tenelsen, V. I. Gavrilenko, and F. Bechstedt, "Optical Properties and Loss Spectra of SiC polytypes from Ab Initio Calculation", Phys. Rev. B, pp1422-1429 (1997).
- [12] C. J. Pickard and M. C. Payne, "Second-order $k \cdot p$ Perturbation Theory with Vanderbilt Pseudopotentials and Plane Waves", Phys. Rev. B, pp4383-4388 (2000).
- [13] Y. R. Shen, "Principles of Nonlinear Optics (Wiley Classics Library)", John Wiley and Sons, Inc.Hoboken, NJ, USA(2003).
- [14] E. Ghahramani, D. J. Moss, and J. E. Sipe, " Full-band-structure Calculation of Second-harmonic Generation in Odd-period Strained (Si)_n/(Ge)_n Superlattices", Phys. Rev. B, pp8990-9002 (1991).
- [15] S. N. Rashkeev, W. R. L. Lambrecht, and B. Segall "Efficient Ab initio Method for the Calculation of Frequency-dependent Second-order Optical Response in Semiconductors", Phys. Rev. B, pp3905-3919 (1998).
- [16] R.D. King-Smith and David Vanderbilt, Phys. Rev. B, 1651 (1993).
- [17] R. Resta, Rev. Mod. Phys. , 899 (1994).
- [18] R. Resta, Ferroelectrics , 51 (1992).
- [19] R. M. Pick, M. H. Cohen, and R. M. Martin, Phys. Rev. B, 910 (1970).
- [20] 物理学辞典 (縮刷版)、物理学辞典編集委員会 (培風館、東京 1992) , p.988.
- [21] D. J. Moss, E. Ghahramani, J. E. Sipe, and H. M. van Driel, "Band-structure Calculation of dispersion and anisotropy in $\chi^{(3)}$ for third-harmonic generation in Si, Ge, and GaAs", Phys. Rev. B, pp1542-1560 (1990).
- [22] D. Vanderbilt, J. Phys. Chem. Solids (2000) 147-151.
- [23] F. Bernardini et al., Phys. Rev. B 56 (1997) R10024.