卒業論文

phonopyによるphonon第一原理計算と精度比較

関西学院大学理工学部

情報科学科 西谷研究室

1513 梁瀬貴生

2015年3月

概 要

phonopy は phonon-dispersion や phonon-DOS，自由エネルギーを第一原理計算ソフトと組み合わせて計算するソフトウェアである．現在，本研究室ではそのような諸計算に商用ソフトの MedeA を使用しているが，そのライセンス維持に巨額の費用がかかる．そこで，フリーのソフトである phonopy への移行を考えている．しかし，phonopy で振動エントロピー等の自由エネルギーの諸計算を行うにあたっては, その精度を検証しておく必要性がある．そこで本研究では phonopy と MedeA を用いて有限温度における自由エネルギーを算出し，その計算結果を比較した．また，研究開始当初には phonopy と MedeA の計算結果が一致しなかったため，Moruzzi らが提案した Quasi-harmonic 近似法を用いてどちらの計算が正しいかを見積も行った．MedeA,Quasi-harmonic 近似法の計算結果が一致したため phonopy の計算を再検討した結果，すべての結果はほぼ一致した．しかし，phonopy で有限温度の自由エネルギーを計算したところ，一定温度の体積変化において極小値を示さないということが明らかとなった．それについて詳しく検討した結果を本論文で報告する.

目 次

第1章　目的

現在，西谷研究室では様々な数値計算ソフトウェアを用いて有限温度における物性について研究を行っている．その際に使用する数値計算ソフト MedeA は高額な維持費を必要とし，計算時間も膨大にかかることから同様の計算を高速で行うフリーソフトの phonopy へ移行を進めている．しかし，phonopy の計算結果が MedeA 同等の精度を算出する確証はない．そこで本研究では phonopy が算出する計算結果の信頼性を検証することを目的とし，有限温度下の物性値を求める際に使用する数値計算ソフト MedeA と同様の計算を短時間かつ効率的に計算を行うフリーソフトウェアの phonopy について精度の検証を行った．

1.1　phonopy

東後篤史が制作したフリーのソフトウェアであり phonon-dispersion や自由エネルギーなどの様々な有限温度の物性を第一原理計算ソフトと連携することで計算することができる．プログラムは主に Python を用いて作成されており，Linux 上で動作する．phonopy の導入に際し，パッケージを操作するコマンドラインツールである APT(Advanced Packaging Tool) を用いるため，本研究ではフリーオペレーティングシステムの Ubuntu を導入した．この Ubuntu をインストールするためには仮想化ソフトウェアパッケージである VirtualBox, 仮想環境の構築から設定までを自動的に行うことができる Vagrant が必要である．導入方法は付録 6 章に記載する．phonopy のサイトへのリンクを記す．http://phonopy.sourceforge.net [4].

1.2　MedeA

データベースと第一原理計算を統合した材料設計支援のためのソフトウェアであり, 構造の検索, 構築, 編集, 計算, 解析までを１つのプラットフォームで行うことができる商用ソ

フトである．MedeA には格子振動に関連する物性を計算するためのツールとして MedeA-Phonon が搭載されており，Job Server/Task Server 機能を使うネットワークを介した計算を行う．現在，西谷研究室では熱膨張を考慮した物性の計算にこのソフトウェアを使用している．MedeA のサイトへのリンクを記す．http://materialsdesign.com [9].

1.3 phonopy と MedeA の計算時間を比較

phonopy は原子座標の対称性を考慮しているため 1 シミュレーションの計算にかかる時間が大きく異なる．表 1.3 にあるように Al を対象とした MedeA の計算では VASP での計算が 191 個，計算時間は約 10 時間ほどかかるのに対し，同様の計算で phonopy は VASP での計算が 1 つ，計算時間は約 15 分ほどで終わることから膨大な計算を必要とする私たちにとって phonopy の方が有益である．しかし，phonopy の結果が必ずしも正しい計算結果を算出しているかは不明であり，その計算の妥当性を検証しなければならない．そこで，振動エントロピー等の自由エネルギーの計算が可能な商用ソフトの MedeA を用いて phonopy と同様の計算を行い，phonopy の計算結果と比較し精度を検証する．

表 1.1: MedeA と phonopy の計算個数と時間比較.

	MedeA	Phonopy
VASP 計算個数	192 個	1 個
総合計算時間	9 時間 49 分	45 分

第2章　計算原理と手法

2.1　有限温度を考慮する計算手法

本章では研究で用いた計算原理と手法について記述する.

2.1.1　第一原理計算を選ぶ理由

第一原理計算とは，原子の持つ基底状態のエネルギーを算出する計算手法である．基底状態とは，熱による振動効果を考慮しない場合のエネルギーであり，最もエネルギーの低い安定な状態のことを言う．基底状態のエネルギーを求めるその他の方法としてはEAMやLJなどがあるが，これらの経験的原子間ポテンシャルに基づく計算手法は計算速度は良いが精度が大きく落ちてしまうので，精確な値を必要とする私たちには適した手法ではない．そこで低速ではあるが，非常に精度の高い第一原理計算を用いる．第一原理計算は系の原子位置を入力として量子力学を支配するシュレディンガー方程式を精確に解き，原子の種類だけから電子構造を求めることで様々な物性を予測する実装となっている.

表 2.1: 第一原理計算とその他の計算のトレードオフ.

	精度	速度
第一原理計算	良い	悪い
EAM,LJ etc...	悪い	良い

2.1.2　VASP(Vienna Ab-initio Simulation Package)

VASPは第一原理計算を行うプログラムである．本研究で必要となる基底状態のエネルギーを算出する手段としてVASPを活用した．VASPの計算の特徴は密度汎関数法による平面波・擬ポテンシャル法を用いている点である．これは原子の内殻電子をのぞいた価電

子だけを考慮する方法であるため，全電子を計算するフルポテンシャル法に比べ比較的高速な計算が可能となる．また，内殻電子は化学結合や物性に影響を与えることが少ないため，擬ポテンシャル法であっても十分な精度で計算を行うことができる．詳しくは本研究室の VASP munual に記載されているためそちらを参照してほしい．

2.1.3 安定した構造の見つけ方

VASP を用いて結晶の基底状態のエネルギーおよび，原子を 1 つ動かしたときに各々の原子にかかる力を求める．このときの動かした原子とかかった力により力の定数を求め，それより周波数 ω と波数の k の関係 (分散関係) を求める．分散関係より，phonon の状態密度を求め，この状態密度から振動自由エネルギーを求める．それに基底状態のエネルギーを加算し，系全体の自由エネルギーを求める．格子の長さを変化させ，以上の過程を繰り返し，同一温度での自由エネルギーを格子の長さについてプロットし，自由エネルギーが極小値をとるときの格子の長さをその温度における計算上の平衡な格子定数とする．図 2.1 の場合，各倍率で体積膨張させた格子定数の温度が 400K 時点のエネルギーをとりだし，縦軸に自由エネルギー [KJ/mol]，横軸に格子定数の膨張倍率を示している．平衡な格子定数は極小値が 1.00 なので，この場合，最も安定している構造は 1.00 倍の格子定数である．

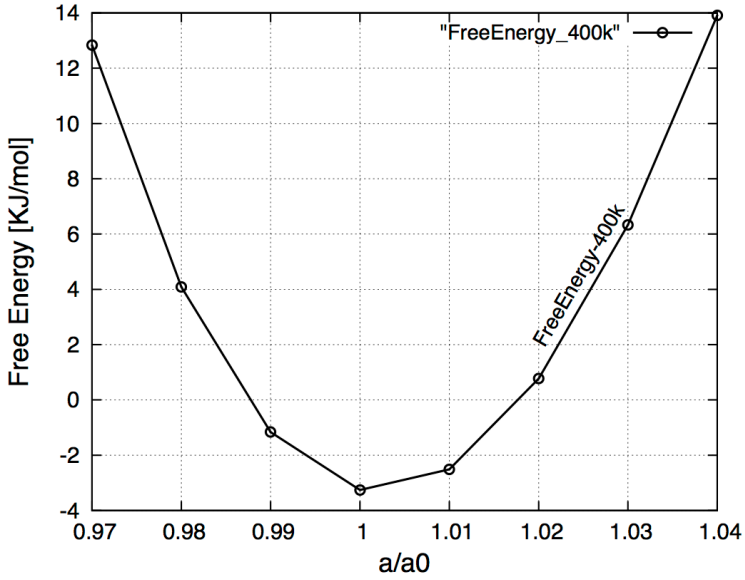

図 2.1: 400K の各格子定数の自由エネルギー．

2.1.4　スーパーセル法による構造拡張

シミュレーション手法の一種であり，単位格子を周期的境界条件を満たすセルとして拡張させる手法である．単位格子を x 軸，y 軸，z 軸にあてはめ各方向に対して倍率を設定することで容易にセルを大きくすることができる．phonopy の場合は原子の座標を示す POSCAR ファイルに対してターミナル上でスーパーセルを構築する．

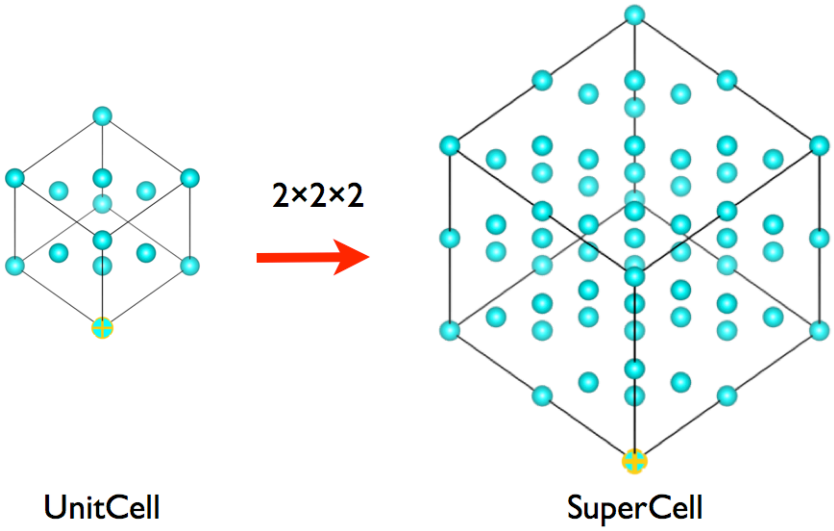

図 2.2: スーパーセル法の模式図.

2.1.5　自由エネルギー

自由エネルギーは,

$$F(a,T) = E(a) + \int_0^\infty D(\omega) f(\omega,T)\, d\omega \tag{2.1}$$

と表される．a は格子の長さ，T は温度を表している．また右辺第一項の $E(a)$ は振動していないときに格子がもつエネルギー，つまり基底状態のエネルギーを表し，右辺第二項

は振動の自由エネルギーを表す．また，右辺第二項の $\int_0^\infty D(\omega)f(\omega,T)\,d\omega$ は Phonon 一個あたりの自由エネルギーを表し，式（2.2）と定義される．

$$F(a,T) = k_{\mathrm{B}}T\ln\left(2\sinh(\frac{h\omega}{4\pi k_{\mathrm{B}}T})\right)d\omega. \tag{2.2}$$

式 (2.2) を用いてある格子定義のもとでの各々での温度における自由エネルギーを求める．ただし，k はボルツマン定数，h はプランク定数を表す．[7]

2.1.6 phonon-DOS 法

phonon-DOS 法は熱が原子に与える振動の影響を考慮することができる計算方法である．phonon とは熱による格子の振動を表したものである．VASP の結果は振動の影響を考慮しておらず，絶対零度の計算となっている．しかし，熱膨張などの諸物性は有限温度においての振動の影響を大きく受けるため，VASP の計算結果から熱による影響を考慮した結果を算出しなければならない．VASP を用いて原子における化学結合をバネモデルと見なし，そのバネ定数を求める．そのバネ定数から波数 k と振動数 ω の分散曲線を算出し，この分散曲線を逆格子空間にわたって積分することにより，phonon-DOS を算出する．phonon-DOS の自由エネルギー関数を式 (2.2) のように積分し，振動自由エネルギーを求める．この自由エネルギーから熱膨張などの諸物性を計算することが可能である．

2.2 Quasi-harmonic 近似法

Quasi-harmonic 近似法とは以下の順序をたどり，有限温度における自由エネルギーを近似的に算出する手法である．今回は 2.2.1 で紹介する数式処理ソフトウェアの Maple を用いて実装した．尚，論文に記載されているプログラムはすべて Maple で作成している．

1. 体積膨張させた結晶の基底状態のエネルギーを VASP を用いて算出し，$E-V$ 曲線を求める．(章 2.2.2 参照，プログラム章 6)

2. 求めた $E-V$ 曲線に結合エネルギーを fitting させ，その極小値から体積弾性率を算出する．(章 2.2.3 参照，プログラム章 6)

3. 体積弾性率から Debye 温度を求め，Debye 関数を定義する．(章 2.2.4 参照，プログラム章 6)

4. 自由エネルギーの式にデバイ関数を代入し，熱振動を考慮した自由エネルギーを算出する．(章 2.2.5, 章 2.2.6 参照，プログラム章 6)

2.2.1 高機能な数式処理ソフト Maple

Maple は，数式処理，視覚化，プログラミング言語など非常に広い領域をカバーしている数式処理ソフトである．したがって，ワープロや表計算ソフトとは違って，プログラム言語を覚えるぐらいの努力が必要である．しかし，Maple スクリプトには C 言語にも類似する手続き型言語，コンパイル作業がいらず，結果を見ながらプログラムを修正できるインタープリタなどの特徴がある．従って BASIC と同等の簡便さで習得できる上に，その適用領域は遥かに広い.[5] 本研究では Maple を用いて quasi-harmonic 近似法を用いた結合エネルギーから自由エネルギーを求めるプログラムを作成する．

2.2.2 2 体間ポテンシャル

バネモデルは，個体のミクロの振る舞いを記述する最も適切なモデルである，個体のバネを数学的に記述 2 体間ポテンシャルは，Lennard-Jones 型と Morse 型があり，本研究では Morse 型を用いた．Morse 型ポテンシャルは，2 原子分子の原子間距離 r に依存する．今回は図 2.3 のように原子間距離を 0.98 から 1.03 倍まで体積膨張させた基底状態のエネルギーからフィッティングを行い，$E-V$ 曲線を作成した．その $E-V$ 曲線に対して近似するような 2 体間ポテンシャルの式を算出する．結合エネルギーを

$$E(r) = a + be^{-\lambda\, r} + ce^{-2\,\lambda\, r} \tag{2.3}$$

と表す．次に Morse 型ポテンシャルに結合エネルギーをフィットさせると以下のようになる．

$$E(r) = A - 2De^{-\lambda\,(r-r_0)} + De^{-2\,\lambda\,(r-r_0)}. \tag{2.4}$$

これに以下の計算式を使うと物性パラメーターが得られる．

$$a = A. \tag{2.5}$$

$$\frac{b}{c} = -2e^{\lambda\, r_0}. \tag{2.6}$$

$$D = \frac{b^2}{4c}. \tag{2.7}$$

これらのパラメーターを使うことで自由エネルギーを求める際に使用する体積弾性率を計算することができる．結合エネルギーの式を立てる中でシミュレーションの整合性を確かめるために必要な実験値の物性エネルギーは Moruzzi の論文を参照してほしい．[8]

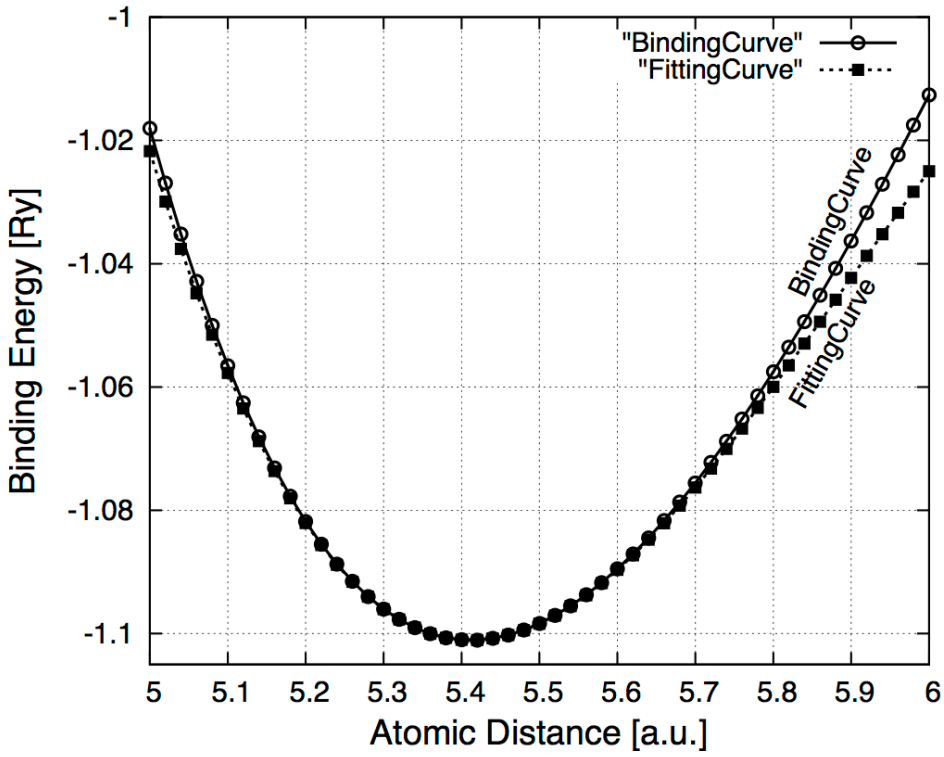

図 2.3: 基底状態と 2 体間ポテンシャルのエネルギー.

2.2.3 体積弾性率 (Bulk modulus)

物体に圧力を加えたときの変形のしにくさを示す指標であり，固体の等方的な固さを表している．これらは体積と圧力の関係から導きだすことができる．

$$x = e^{-\lambda\, r} \tag{2.8}$$

とすると結合エネルギーの式 (2.3) は

$$E = a + bx + cx^2 \tag{2.9}$$

と表される．熱膨張を計算するには圧力の体積及び温度依存性を決定する必要がある．そこで圧力 P は全エネルギーから派生した負の体積をとることで評価することができる.[6]

$$P = -\frac{\partial\, E}{\partial\, V}. \tag{2.10}$$

この式を以下のように変形させる

$$P = -\frac{\frac{\partial\, E}{\partial\, x}}{\frac{\partial\, V}{\partial\, x}}. \tag{2.11}$$

また体積 V は

$$V = \frac{4}{3}\pi\, r^3 \tag{2.12}$$

と表される．また式 (2.8) より r は

$$r = -\frac{\ln x}{\lambda} \tag{2.13}$$

と表されることから体積 V は

$$V = \frac{4\ \pi}{3}\left(\frac{-\ln x}{\lambda}\right)^3. \tag{2.14}$$

この式から x の関数の見なすことが出来る．これを式 (2.11) に戻すと圧力 P は以下のように定義できる．

$$P = \frac{x\ \lambda^3}{4\ \pi \ln x^2}\left(b + 2cx\right). \tag{2.15}$$

ここで，体積弾性率 B は

$$B = -V\frac{\partial\ P}{\partial\ V} \tag{2.16}$$

と定義される．なので式 (2.15) と同様に

$$P = -V\frac{\frac{\partial\ P}{\partial\ x}}{\frac{\partial\ V}{\partial\ x}}. \tag{2.17}$$

式 (2.17) の式のように変形を行い式 (2.16), 式 (2.15) を式 (2.17) に代入すると体積弾性率は

$$B = -\frac{x\ \lambda^3}{12\ \pi\ \ln x}\left((b+4cx) - \frac{2}{\ln x}\,(b+2cx)\right). \tag{2.18}$$

上記の式に必要な a,b,c は結合エネルギーの式から算出され，図 2.4 のようになる この体

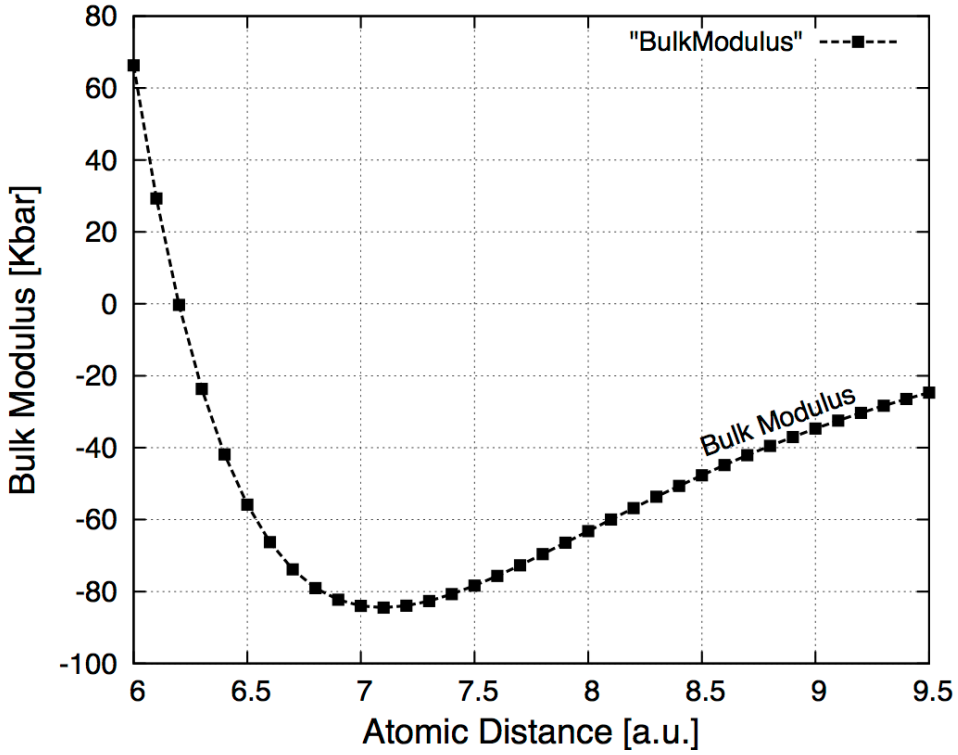

図 2.4: 原子間距離の変化に伴う体積弾性率の変化の様子.

積弾性率は次項で記述するデバイ温度 Θ_{D} を求める際は必ず必要となってくる．ここで注意しなければならない点は体積弾性率の単位を [Ry/a.u.^{3}] から [eV/A^3],[GPa],[kbar] と変換させることである，詳しくは付録の Quasi-harmonic プログラムを参照してほしい.

2.2.4 デバイ温度 (Debye Temperature)

デバイ温度とは格子振動や，結晶の結合力といったような物理量と深くかかわり合いを持つ温度のことを指し，物質の硬さの指標となるそれぞれの物質の定数である．次元は温度と同等で一般的に，硬い物質ほど高く逆に柔らかい物質ほど低い値を取る．低温では低振動数の格子モードのみが基底状態から励起されることになる．低振動数のモードとしては，長波長の音響モード，つまり音波があり，その分散関係は

$$\omega = v_s k \tag{2.19}$$

の形をとる．ここで v_s は音速を表す．この分散関係に伴う状態密度関数 $g(\omega)$ は，

$$g(\omega) = \frac{V k^2}{2\,\pi^2} \frac{dk}{d\,\omega} = \frac{V\,\omega^2}{2\,\pi^2 v_s^3} \tag{2.20}$$

と表すことができる．ここでは v_s は伝搬の方向に依存し，ω は k の大きさだけによるという仮定は成立しない．しかし，もし因子 $1/v_s^3$ が全ての方向にわたって平均したものを表していると見なすと，

$$g(\omega) = \frac{V\,\omega^2}{2\,\pi^2} \left(\frac{1}{v_l^3} + \frac{2}{v_t^3} \right) \tag{2.21}$$

と表すことができる．ここでの状態密度は 3 個の音響分岐，すなわち 1 個の縦波 (longitudinal) と 2 個の横波 (transverse) にわたって和をとる必要がある．ここで，v_t，v_l はそれぞれ縦波と横波のモードの音速であり，

$$\frac{1}{v_s^3} = \frac{1}{v_l^3} + \frac{2}{v_t^3} \tag{2.22}$$

これを解くと，

$$\frac{V}{6\,\pi^2} \left(\frac{1}{v_l^3} + \frac{2}{v_t^3} \right) \omega_{\mathrm{D}}^3 = 3 \tag{2.23}$$

と表される．ここで 3 個の音響分子の平均を

$$\frac{3}{v^3} = \frac{1}{v_l^3} + \frac{2}{v_t^3} \tag{2.24}$$

とし，式（2.23）に代入すると，

$$\frac{V}{6\ \pi^2}v^{-3}\omega_{\mathrm{D}}^3 = 1 \tag{2.25}$$

と変形することができる．また，v を ρ (密度) と B(体積弾性率) で定義すると，

$$v = \sqrt{\frac{B}{\rho}} \tag{2.26}$$

となる．また ρ は

$$\rho\ = \frac{M}{V} \tag{2.27}$$

と表され，体積 V は，

$$V = \frac{4}{3}\pi\ r^3 \tag{2.28}$$

だから式（2.27）式（2.28）を式（2.26）に代入すると

$$v = \sqrt{\frac{4}{3}\pi\ r^3 B} \tag{2.29}$$

というように変形できる．これを式（2.25）に代入すると，

$$\omega_{\mathrm{D}}^3 = 6\ \pi^2\frac{3}{4\ \pi\ r^3}v^3 = 6\ \pi^2\left(\frac{4}{3}\pi\ r^3\right)^{-1}\left(\frac{4\ \pi\ r^3}{3}\right)^{\frac{3}{2}}\left(\frac{B}{M}\right)^{\frac{3}{2}} \tag{2.30}$$

となり，

$$\omega_{\mathrm{D}} = (6\ \pi^2)^{\frac{1}{3}}\left(\frac{3}{4}\pi\ r^3\right)^{\frac{1}{6}}\sqrt{\frac{B}{M}} \tag{2.31}$$

と表すことができる．ここで，デバイ温度は，

$$k_B\Theta_{\mathrm{D}} = \frac{h}{2\ \pi}\omega_{\mathrm{D}} \tag{2.32}$$

と一般的に表され，このときの k_{B}，ω_{D} はそれぞれボルツマン定数 k_{B}，プランク定数 h と呼ばれる定数のことを指す．また，ω_{D} はデバイ振動数である．これに式（2.31）を代入すると

$$\Theta_{\mathrm{D}} = \frac{h}{2\ \pi\ K_B}(6\ \pi^2)^{\frac{1}{3}}\left(\frac{3}{4}\pi\ r^3\right)^{\frac{1}{6}}\sqrt{\frac{B}{M}}. \tag{2.33}$$

これを計算すると

$$\Theta_{\mathrm{D}} = 67.48\sqrt{\frac{rB}{M}} \tag{2.34}$$

となる．ここで，v_t，v_l をそれぞれ ρ を用いて表すと，

$$v_t = \sqrt{\frac{S}{\rho}} \tag{2.35}$$

$$v_t = \sqrt{\frac{L}{\rho}} \tag{2.36}$$

というように表され，これらに Moruzzi らが実験値より計算した値，$S = 0.30B$，$L = 1.42B$ をそれぞれ代入すると，v は

$$v = 0.617\sqrt{\frac{B}{\rho}} \tag{2.37}$$

となる．これを式（2.31）に代入し，Θ_{D} を計算すると，

$$(\Theta_{\mathrm{D}})_0 = 41.63\sqrt{\frac{r_0 B}{M}} \tag{2.38}$$

となる．$r0$ は平衡原子間距離 [a.u.]，M は原子量を表し，B は $r0$ において計算された体積弾性率 [kbar] となる．[6] 式 2.38 は図 2.5 のようにグラフ化され，原子間距離が大きくなるにつれてデバイ温度が下がる．

2.2.5 デバイ関数

デバイ関数とはデバイ温度をパラメータとする関数であり以下のように表される．

$$D(y) = \frac{3}{y^3}\int_0^y \frac{e^x x^4}{(e^x - 1)^2} dx. \tag{2.39}$$

これは y が 0 に近づくと，D は 1 に近づくので，古典統計力学に対応している． この関数の $y = \Theta_{\mathrm{D}}$ として計算していく．[6] 以下の図 2.6 は 300K のときのデバイ関数と原子間距離の関係を表している．

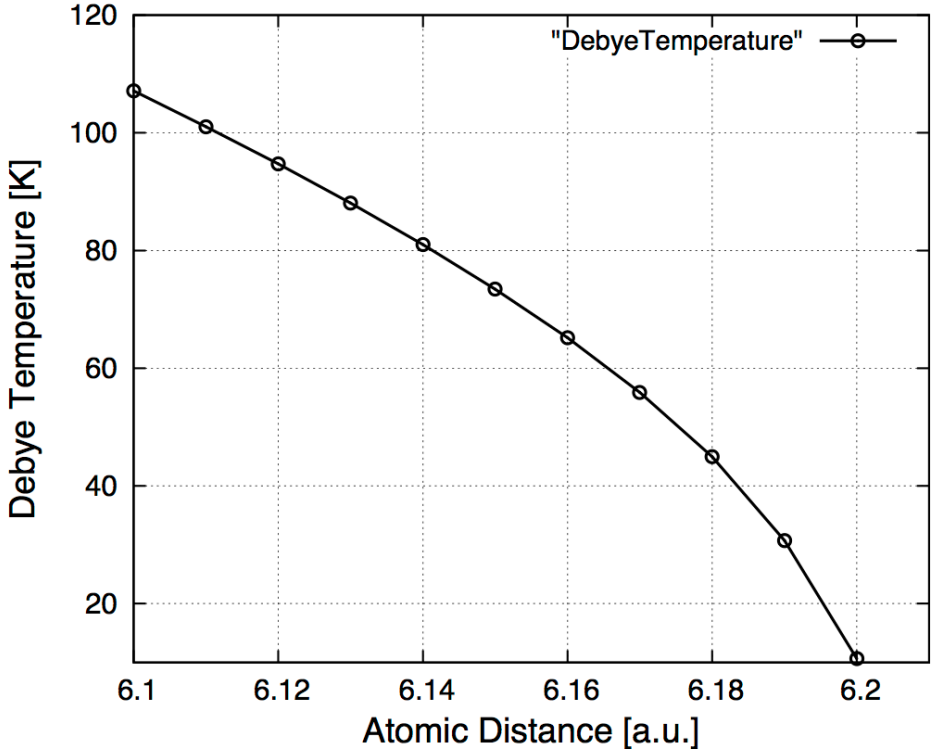

図 2.5: デバイ温度の原子間距離依存性.

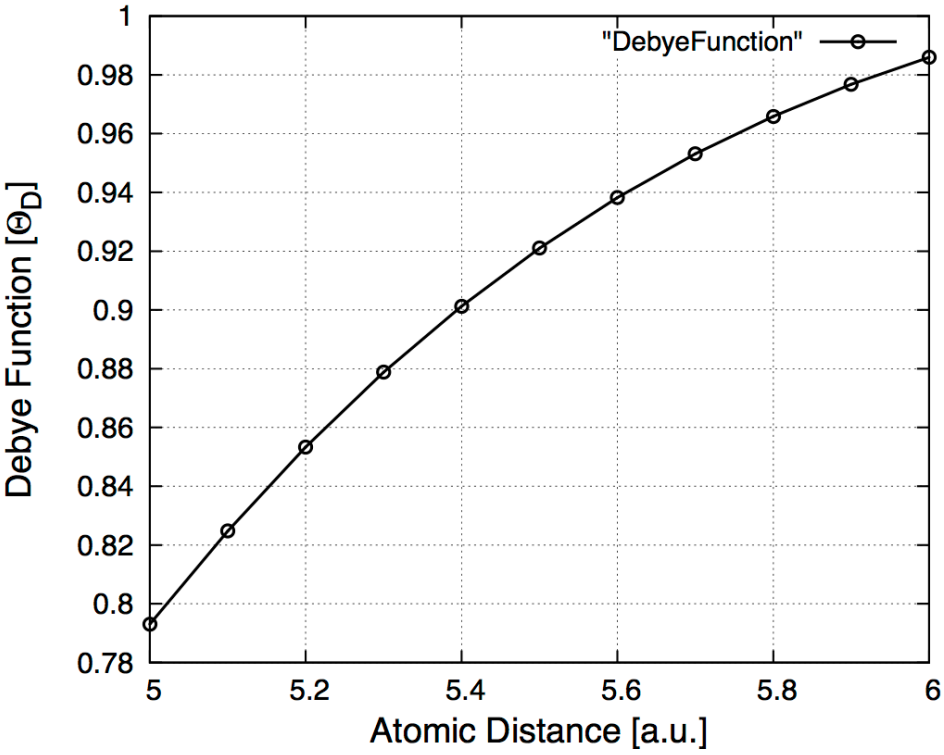

図 2.6: デバイ関数と原子間距離依存性.

2.2.6　デバイ関数を導入した自由エネルギー

前章でも述べたように自由エネルギーとは，振動している物質の全エネルギーである，以下の自由エネルギーの式に対してデバイ関数を導入する．

$$F(r,T) = E(r) + E_D(r,T) - TS_D(r,T). \tag{2.40}$$

ここで，T は温度，E_D と S_D はデバイ関数で，E_D と S_D はそれぞれ

$$E_D(r,T) = 3k_BTD\left(\frac{\Theta_\mathrm{D}}{T}\right) + E_0 \tag{2.41}$$

と，

$$S_D(r,T) = 3k_B\left[\frac{4}{3}D\left(\frac{\Theta_\mathrm{D}}{T}\right) - \ln\left(1 - e^{-\frac{\Theta_\mathrm{D}}{T}}\right)\right] \tag{2.42}$$

となる．ここで E_0 は零点エネルギーを表し，

$$E_0 = \frac{9}{8}k_B\Theta_\mathrm{D} \tag{2.43}$$

とされる．以上より自由エネルギーの最終形態として式をまとめると，以下のようにまとめることができる．

$$f(r,T) = E(r) - k_BT\left[D\left(\frac{\Theta_\mathrm{D}}{T}\right) - 3\ln\left(1 - e^{-\frac{\Theta_\mathrm{D}}{T}}\right)\right] + \frac{9}{8}k_B\Theta_\mathrm{D}. \tag{2.44}$$

これまで考慮してきた，体積弾性率，デバイ温度，デバイ関数をこの自由エネルギーの式に当てはめることで，容易に物質の自由エネルギーを計算することができ，さらに熱膨張も計算することができるようになる．

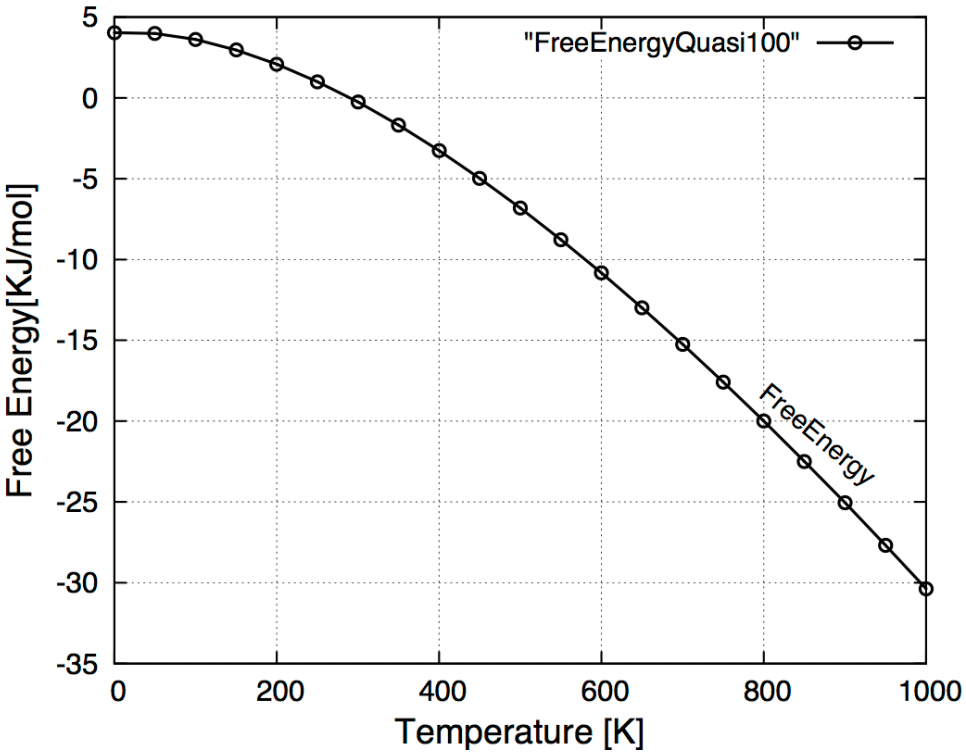

図 2.7: 熱膨張による平衡原子間距離の変化.

第3章　手順比較

ソフトウェアの性能を比較する際は計算時間だけでなく，インターフェースや作業の効率性を比較する必要がある．そこで本章では phonopy と MedeA で同一の作業を行うときの手順を比較する．

3.1　モデルの構築の比較

計算対象となるモデルの構築を行うにあたって，MedeA と phonopy では違った方法を用いている．MedeA は Windows 上で GUI を用いてモデルを構築する．計算対象となる結晶モデルの元素記号や格子定数を設定し，モデルの編集を行う．phonopy はあらかじめ用意しておいたユニットセルファイルからスーパーセルを作成する．以下に両方法の手順を示す.

MedeA でモデル構築を行う手順

1. メインメニューの New bulk System を選択し，周期境界を示すボックスを表示.
2. Edit タブの Edit structure を選択し，Spacegroup を設定することで対称性を考慮する．Al の場合，Spacegroup は Fm-3m となる
3. 格子定数の設定を行うため各軸の長さを入力する．軸の長さには VASP を用いて構造緩和を行った格子定数を使用する．
4. Add Atom タブを選択し，計算対象となる原子の種類と原子座標を入力する
5. 原子の種類が 2 つ以上あるセルを構築するには 3 からの作業を原子数分の回数行う
6. Builders タブから Build supercells を選択し，ユニットセルをスーパーセルに拡張する．

phonopy でモデル構築を行う手順

1. 計算対象となる POSCAR を用意し，VASP を用いて構造緩和を行う．

2. phonopy を用いてスーパーセルをコマンド 1 つで作成する．作成方法は章 6 を参照．

モデル構築の手法を比べると MedeA は原子の位置から対称性の考慮までを行う必要があるのに対し，phonopy はユニットセルさえあれば容易にモデルを構築できる．MedeA の作業にくらべ，比較的少ない作業数でモデルを構築できるため，入力のミスによる計算誤差を少なくする事が出来る．ただし，phonopy だけでは視覚的に構造の整合性を確認する事が出来ないため，視覚化ソフトと組み合わせて使用する必要がある．

3.2 計算設定の比較

基底状態のエネルギーの計算には MedeA,phonopy どちらも VASP を用いるため，どちらも INCAR の設定が計算の設定となる．計算設定の方法は INCAR の編集環境が GUI と CUI という違いを持つ．MedeA には図 3.1 のような VASP の計算エンジンがあり，インターフェースが用意されている．これらを用いる事で INCAR を編集する事が出来る．その他の構造緩和の設定や DOS, バンド構造の出力もボタン一つで設定が可能になっている．

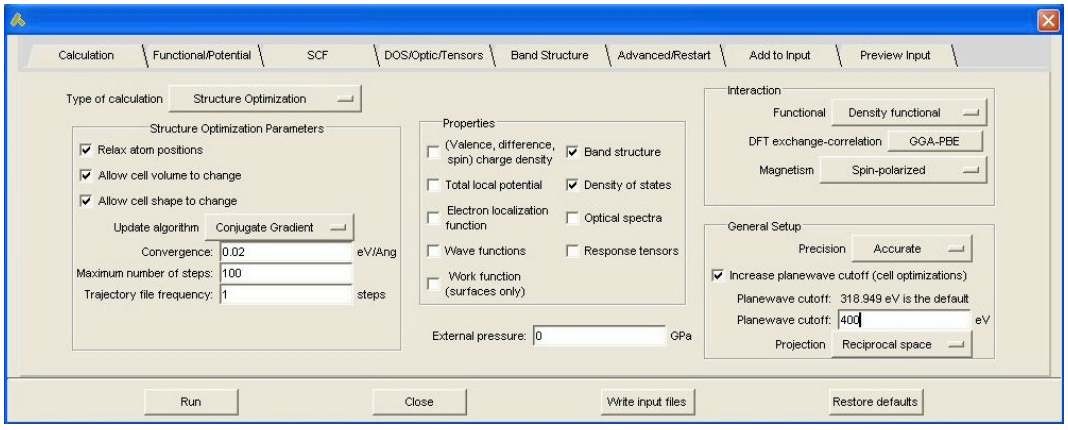

図 3.1: MedeA の計算設定画面

phonopy では以下のように INCAR ファイルを直接編集する必要がある．与えられたパラメータの値を変更することで計算目的に合った条件を設定する．

INCAR ファイル

```
    PREC = Accurate
   ENCUT = 300
  IBRION = 2
     NSW = 60
    ISIF = 3
    ALGO = V
    NELM = 60
  NELMIN = 12
  NELMDL = -12
   EDIFF = 1.0e-05
  EDIFFG = -0.02
 VOSKOWN = 1
   ISPIN = 1
  INIWAV = 1
  ISTART = 0
  ICHARG = 2
   LWAVE = .FALSE.
  LCHARG = .TRUE.
 ADDGRID = .TRUE.
   LREAL = Auto
```

MedeA の計算設定も GUI で入力した値を最後に INCAR をみて確認するので，直接操作が可能な phonopy の方が MedeA と比べ，利便性が高い．

3.3 計算実行

MedeA では phonon 計算の場合も基底状態のエネルギーの計算と同様，パラメーターを編集するインターフェースが用意されている．また，JobServer/TaskServer などのジョブ管理システムが導入されており，計算の管理を行うことができる．phonopy では VASP の計算結果をもとに phonon 計算を行う．実行はコマンドを用いて行うことができ，各計算の設定はタグを用いて行うことができる．

phonopy の実行方法

1. VASP を用いて，微小変位させた原子配置の個数分だけ基底状態のエネルギーを計算する．
2. 作成された vasprun ファイルを計算対象の POSCAR があるフォルダに移動させ，phonon を算出する，章 6 を参照.

MedeA の実行方法

1. VASP を用いて構造緩和をおこなった原子を選択し，MedeA-phonon を起動させる
2. スーパーセルを設定し，JobServer に計算を投げる．その後は自動に計算を行う．

第4章　結果

本章では Al,NaCl の自由エネルギーを phonopy,MedeA を用いて算出し，その精度を Quasiharmonic 近似法を用いて検証した結果を記す．また，計算で用いた原子の格子定数を表 4.1 にまとめた．これらの格子定数は VASP で構造緩和をおこなった安定な格子定数である．構造緩和の手法として原子は位置をそれぞれ移動させて緩和する内部緩和と格子定数を変化させ緩和する外部緩和のどちらも行う方法を用いた.

表 4.1: 計算に用いた格子定数.

	Al	NaCl
体積膨張なし (× 1.00)	4.0371999	5.6372496
体積膨張あり (× 1.03)	4.1583159	5.8063671

4.1　phonopy の動作

図 4.1 が phonopy を用いて Al の有限温度下における自由エネルギーの変化，図 4.2 が NaCl に同様の計算を行った結果であり，縦軸に自由エネルギー [KJ/mol], 横軸に温度 [K] を表している．図中の NomalCell は体積膨張を考慮せずに行った計算，VolumeExpansionCell は体積膨張を考慮し格子定数を 1.03 倍させた計算結果である．Al のグラフはお互いに交わることなく，体積膨張した原子の方が低い値をとり，温度が上がるにつれて体積膨張なしの原子と体積膨張をしている原子とのエネルギーの差が大きくなった. また NaCl の自由エネルギーは Al に比べて負に大きい値を示した.

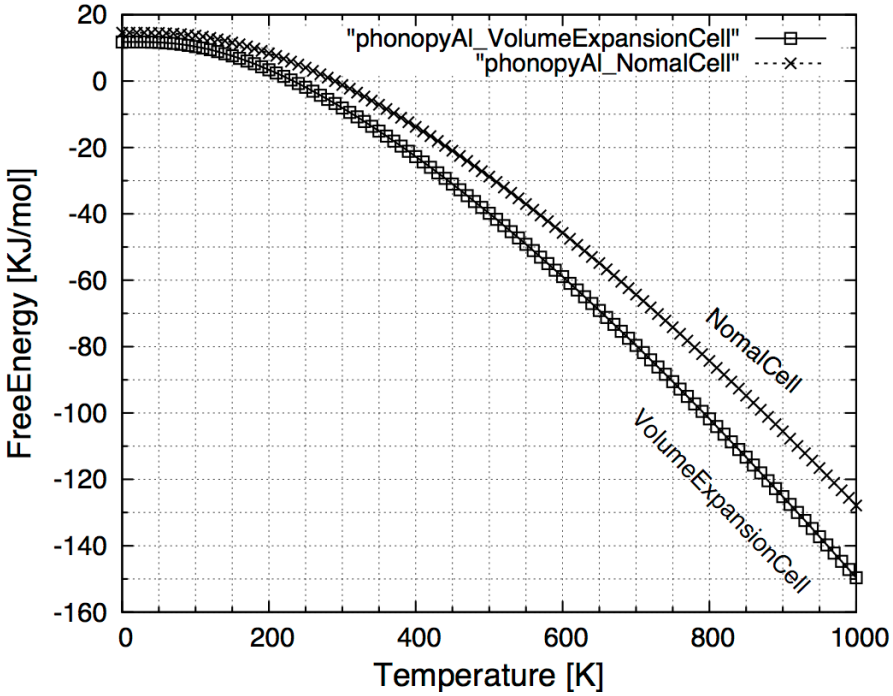

図 4.1: phonopy による Al の有限温度下の自由エネルギー.

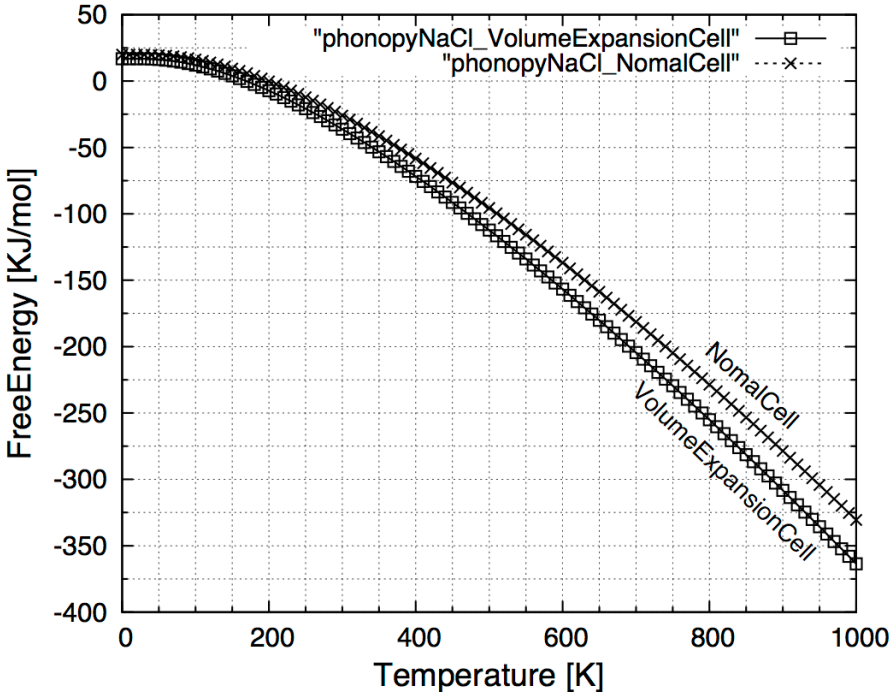

図 4.2: phonopy による NaCl の有限温度下の自由エネルギー.

4.1.1 phonopy で最安定な構造を調べる

適当な温度を選択し，各格子定数の倍率ごとに自由エネルギーの様子をプロットすることで極小値がわかる．図 4.3 は縦軸を自由エネルギー [KJ/mol], 横軸を体積の膨張率として表しており，極小値における膨張率が最安定な格子定数となる． しかし，300K における各軸比ごとに自由エネルギーを算出すると極小値が phonopy で見られず，体積が大きくなるにつれて自由エネルギーの値は下がっていった．

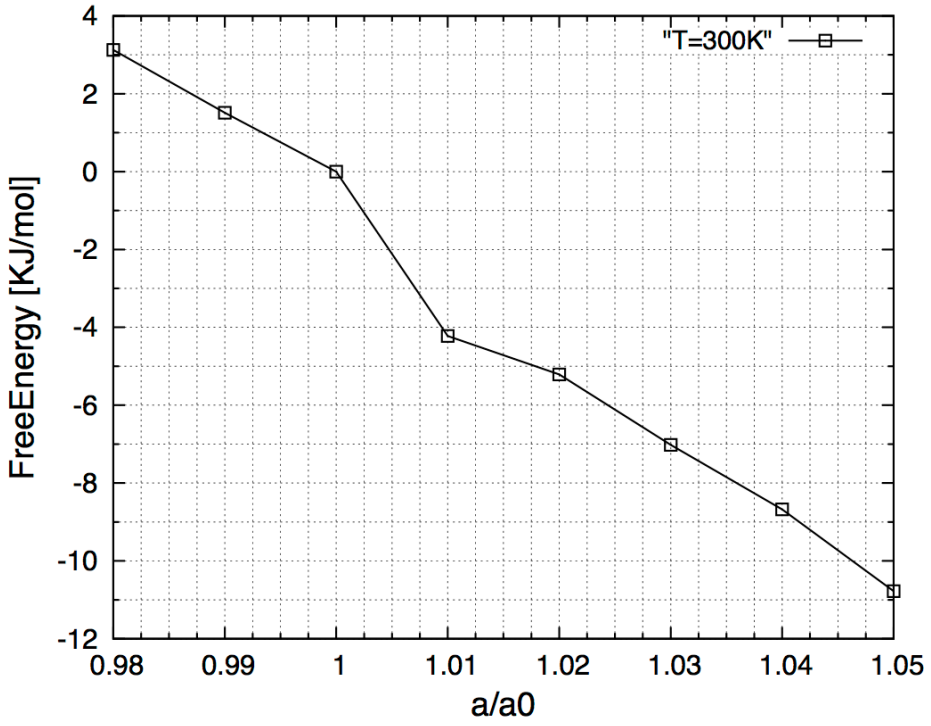

図 4.3: phonopy を用いた 300K における Al の自由エネルギー.

4.2 MedeA の動作

図 4.4 が MedeA を用いて Al の有限温度下における自由エネルギーの変化，図 4.5 が NaCl に同様の計算を行った結果であり，縦軸に自由エネルギー [KJ/mol], 横軸に温度 [K] をとる．図 4.5 の NomalCell は体積膨張を考慮せずにした結果，VolumeExpansionCell は体積膨張率を考慮し 1.03 倍に膨張させたときの結果である．Al は図 4.4 のように温度が約 700K の時点でグラフが交差し上下が入れ替わった．

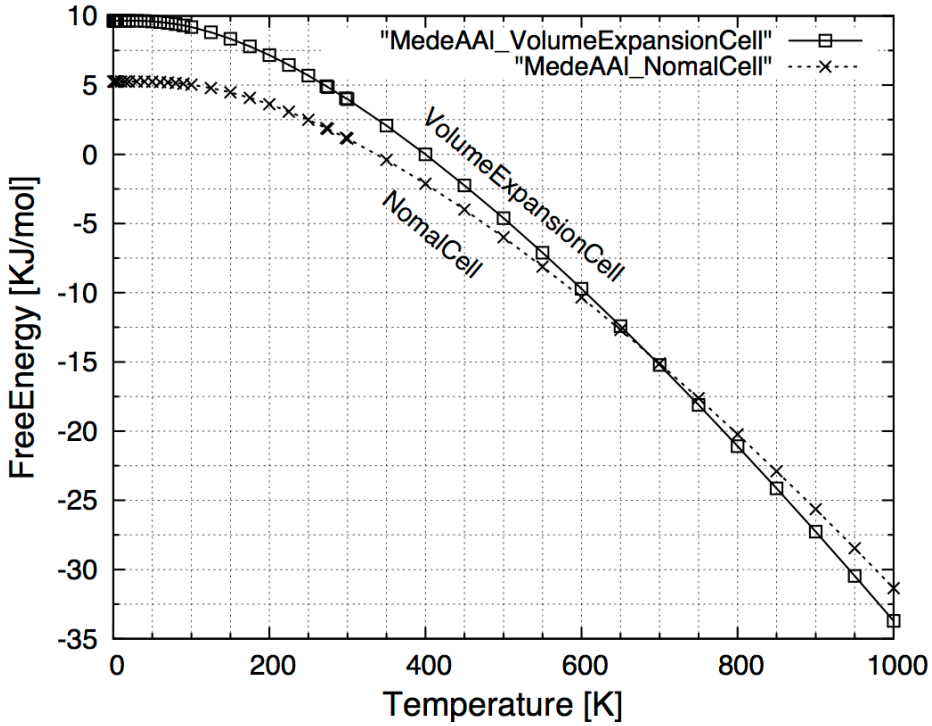

図 4.4: MedeA による Al の有限温度下の自由エネルギー.

しかし，NaCl は図 4.5 のようにグラフが交わることなく温度が上昇するにつれて値が落ちていき，体積膨張した原子は体積膨張を考慮しない原子に比べて低い値をとった．また，Al と比較するとエネルギーが大きく下回る結果となった．これは，NaCl のイオン結合による影響を考慮できていないため自由エネルギーを上手く再現できなかっと考えられる．

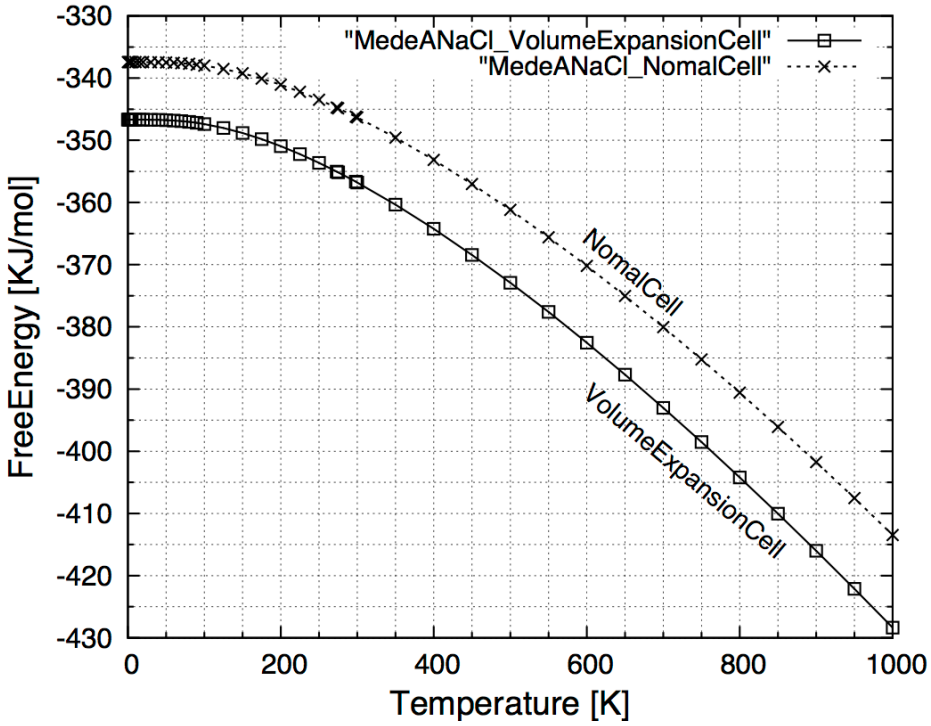

図 4.5: MedeA による NaCl の有限温度下の自由エネルギー.

4.2.1 MedeA で最安定な構造を調べる

図 4.6 のように Al の温度 100K,300K,500K における各軸比ごとに自由エネルギーの様子をプロットすると MedeA では極小値が見られた．Caluclation-data は MedeA による計算値であり，実線は計算値に対して fitting を行ったグラフである．また，極小値の軸比は温度が上昇するにつれて大きくなっていることから熱膨張の影響を再現することが出来ている．

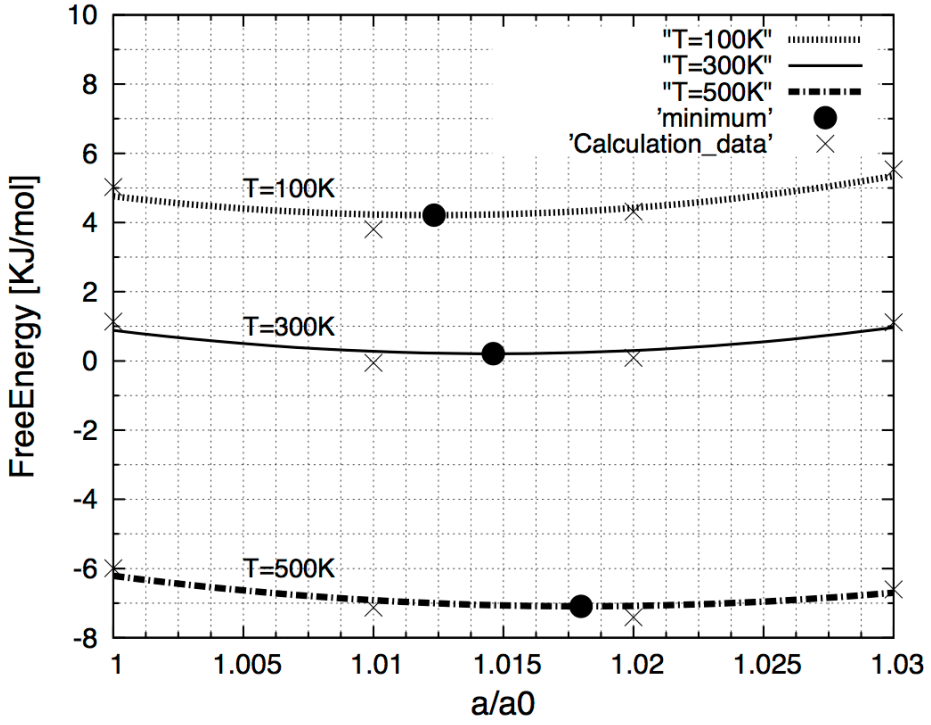

図 4.6: MedeA を用いた温度上昇による原子間距離の伸張の様子.

4.3 Quasi-harmonic 近似法の動作

自由エネルギーを体積弾性率から求める Quasi-harmonic 近似法を用いて，有限温度下における自由エネルギーを算出した．このプログラムは各膨張率における基底状態のエネルギーを入力とし，有限温度の自由エネルギーを図 4.7 のように出力する．NomalCell が体積膨張なしの原子，VolumeExpansionCell が 1.01 倍に体積膨張をさせた原子の結果を表している．温度が上がるにつれて体積膨張させた原子は体積膨張なしの原子よりも自由エネルギーの低下が大きく見られ，約 900K 前後ではわずかだが体積膨張なしの原子よりも自由エネルギーの値を低くとっていることが分かる．

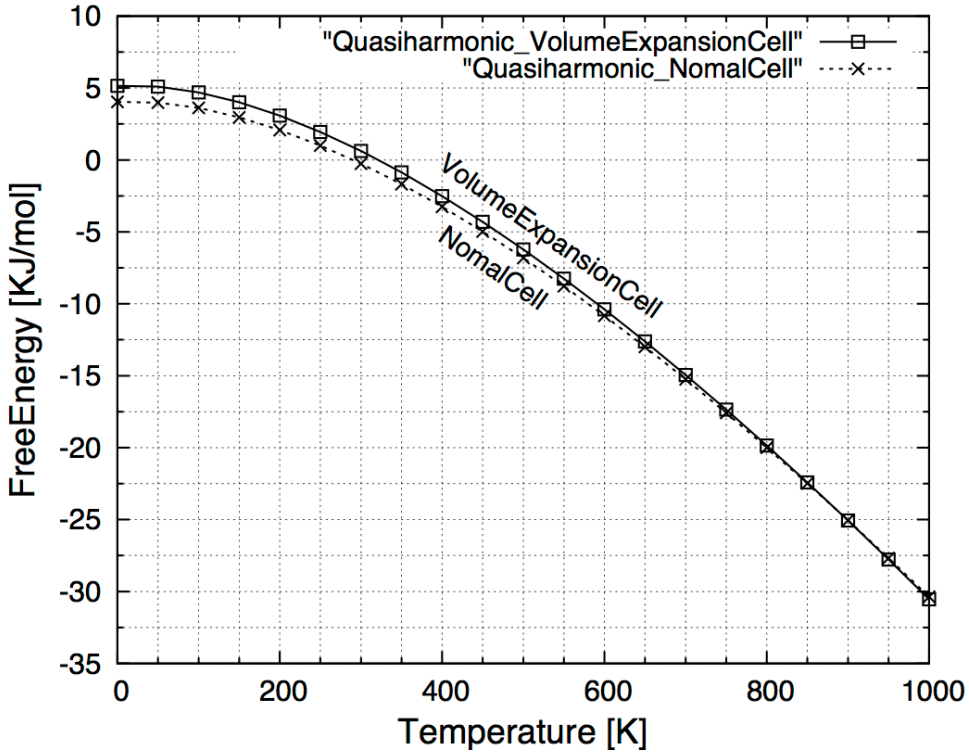

図 4.7: Quasiharmonic 近似法による Al の有限温度下の自由エネルギー.

4.3.1 Quasi-harmonic 近似法で最安定な構造を調べる

図 4.8 は横軸を原子間距離 [a.u.]，縦軸を自由エネルギー [KJ/mol] として，温度毎の自由エネルギーを表している．この自由エネルギーの極小値にあたる部分が原子の引力と斥力が釣り合う平衡原子間距離であり，グラフでは minimum で表されている．温度が上昇するにつれて平衡原子間距離が大きくなっていることから熱膨張によるエネルギーの変化があることが分かり，熱膨張の影響が正しく再現されていることが分かる

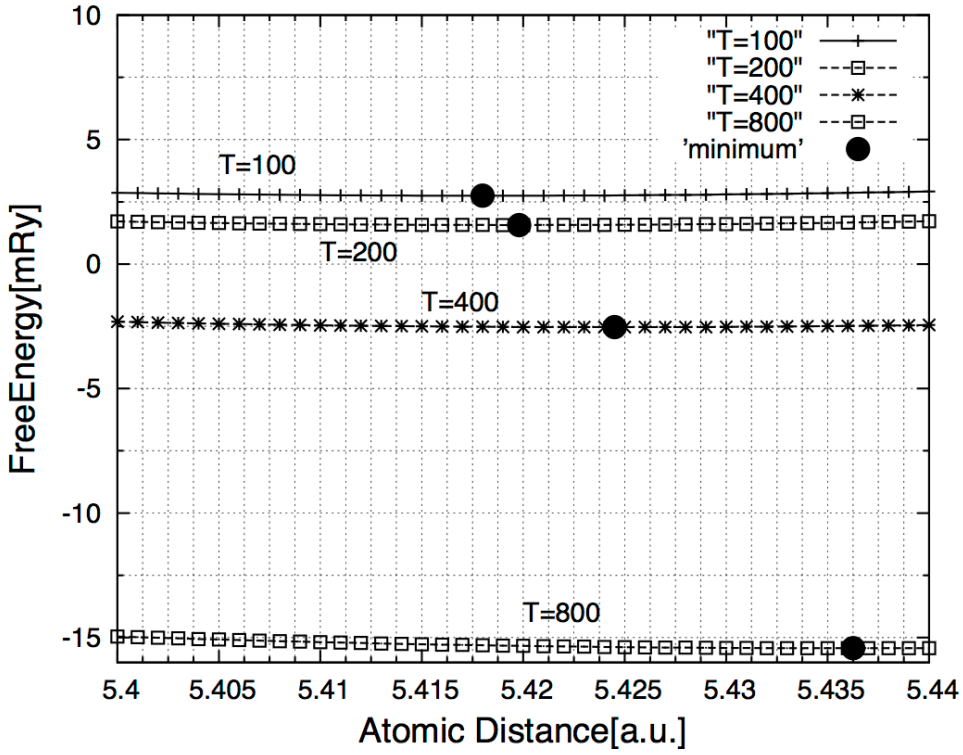

図 4.8: 温度上昇による原子間距離の伸張の様子.

第5章　考察

5.1　結果比較

5.1.1　温度変化による自由エネルギーの様子

phonopy,MedeA,Quasi-harmonic 近似法によって算出した Al の自由エネルギーの比較を行った．図 5.1 は縦軸は自由エネルギー [KJ/mol]，横軸は温度 [K] であり，各手法の温度変化における自由エネルギーの変化を表している．phonopy の自由エネルギーは，MedeA の自由エネルギーの値とほぼ一致していることから phonopy は MedeA 同等の計算結果を再現していることが分かる．また，熱膨張の影響を再現している Quasi-harmonic 近似法の計算結果ともほぼ一致することから，phonopy の算出した自由エネルギーの絶対値は信頼のおける結果ということが明らかとなった．しかし phonopy の算出結果には大きな問題点がある．その問題点について次節で解説する．また，phonopy はセル全体の自由エネルギーを算出しているため，他の計算結果と比較する際は値をすべて 1 原子あたりに換算する点を注意しなければならない．

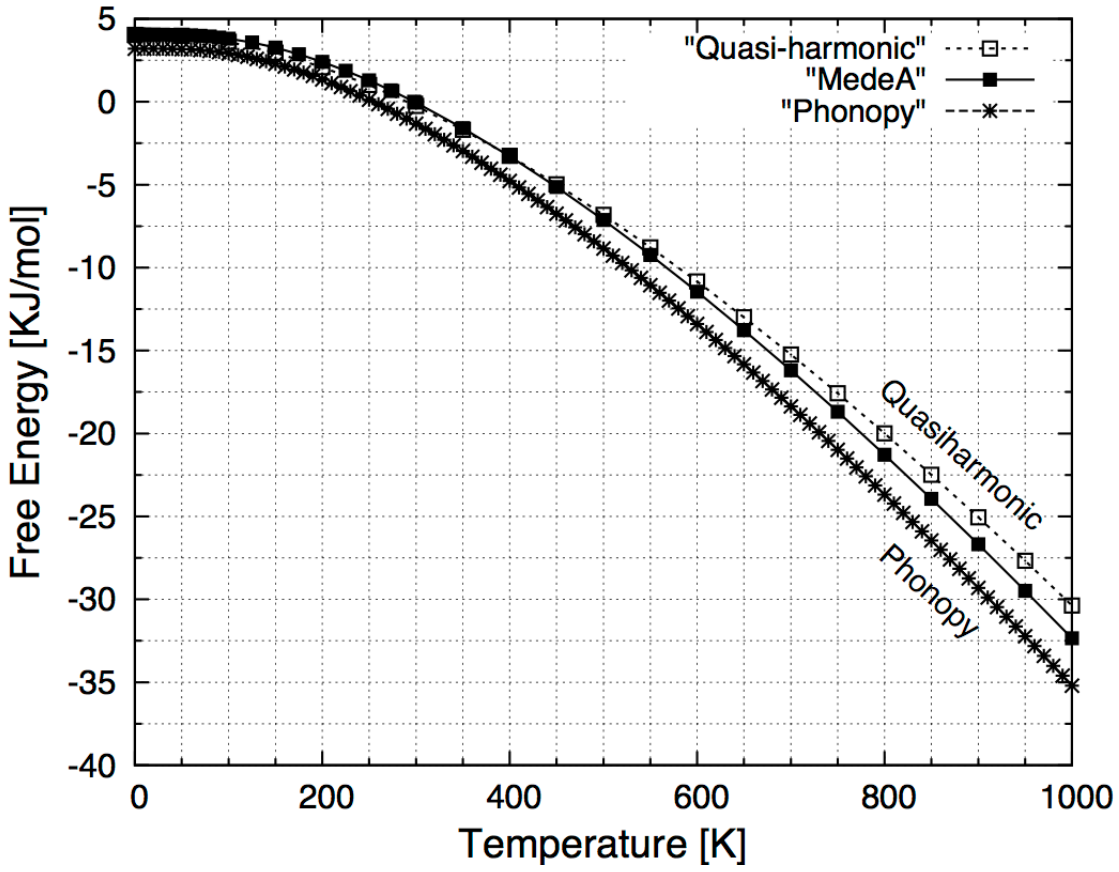

図 5.1: それぞれの手法を用いた Al の有限温度下の自由エネルギー.

5.1.2　体積変化による自由エネルギーの様子

phonopy,MedeA,Quasi-harmonic 近似法を用いて Al が 300K のときの最安定な格子定数を算出した．図 5.2 は縦軸に自由エネルギー [KJ/mol], 横軸に格子定数を体積変化させた倍率を示している．Calculation-data は MedeA と phonopy の計算値であり，実線は計算値に対して fitting を行ったものである．MedeA,Quasi-harmonic 近似法は極小値を持つが phonopy は極小値がなく値は温度が上がるにつれて自由エネルギーの値は低下している．このことから phonopy で算出した自由エネルギーは最安定をとることが出来ないということが明らかとなった．最安定がとれなければ熱膨張係数を算出することが出来ず，これ以上の計算は不可能である．従って phonopy の算出する結果は正しく再現出来ているのは絶対値のみであり，熱膨張の影響は考慮できていないということが分かった．

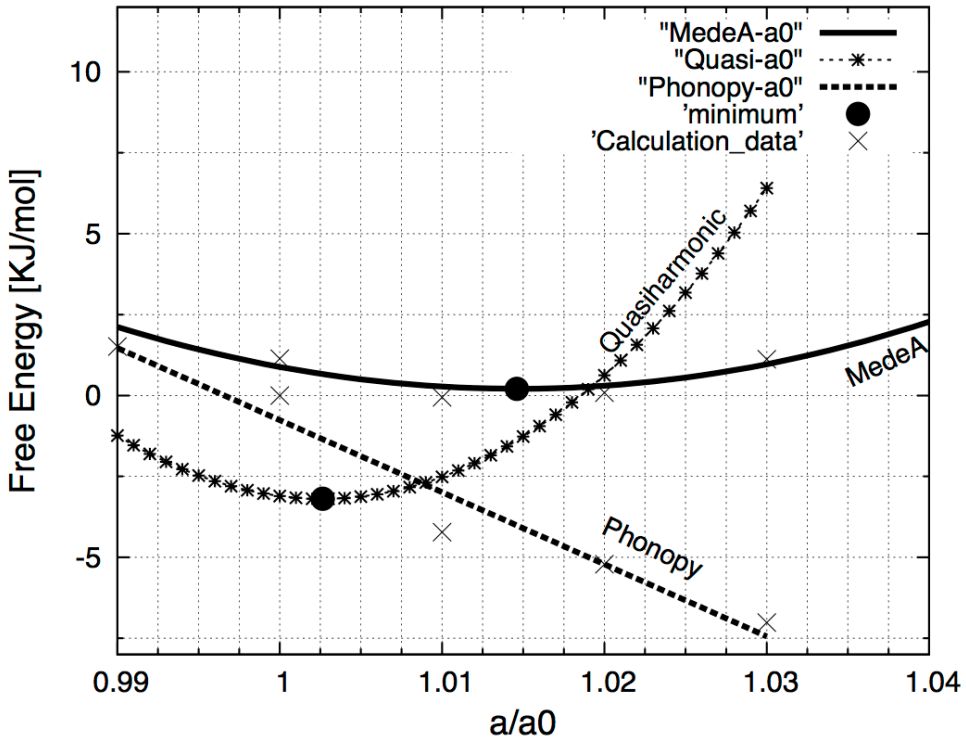

図 5.2: 各手法における極小値の有無.

第6章　総括

本研究ではphonopyとMedeAを用いてAl,NaClの有限温度における自由エネルギーを算出し，お互いの計算結果を比較することでphonopyの計算精度を検証した．研究開始当初はphonopyとMedeAの結果が一致しなかったため,Moruzzi らが提案したQuasi-harmonic近似法を用いてAlの自由エネルギーを算出することで，どちらの計算が正しいかを見積もった．　その結果,MedeAとQuasi-harmonic近似法の算出した値は一致し，phonopyの計算結果を再度検討することになった．計算を見直し，再検討した結果，phonopyとMedeA,Quasi-harmonic近似法の自由エネルギーの絶対値はほぼ一致した．phonopyとMedeAの結果が違った原因としてphonopyは自由エネルギー値を１ユニットセル毎に算出しているため，1原子あたりに算出しているMedeAと比較する際はphonopyの値を1原子あたりのエネルギーに変更しなければいけないことが分かった．しかし，同一温度で各格子定数ごとの自由エネルギーの様子を比較すると，MedeA,Quasi-harmonic近似法では最小値を得られたが，phonopyでは値は軸比が上がる毎に自由エネルギーが低下し，極小値は得られなかった．この結果から熱膨張係数などの熱膨張の影響を考慮した物性値の計算に移行することは困難であり，phonopyはMedeA同等の計算結果を再現できないという事が明らかとなった．

謝辞

本研究を遂行していくにあたり，終始多大なる有益なご指導，また私の就職後の事についても親身になって考えてくださいました西谷滋人教授に深い感謝の意を表します．研究を進めるにあたり，多くの援助，親身に相談にのって頂きました清原資之先輩をはじめ，さまざまな知識の供給などの御協力を頂いた西谷研究室の皆様，そして私が無事に卒業できるまで，支えてくださった両親に感謝の意を表します．最後になりましたが，この場を借りて心から深く御礼申し上げます．

付録

phonopy の導入と使用方法

インストール方法

以下の手順で環境を構築し，phonopy のインストールを行う．VirtualBox,Vagrant,UbuntuOS のインストールは phonopy をインストールする際に使用する apt-get ツールを導入するために行うので，既存で APT がある場合は省略してもかまわない．

VirtualBox

VirtualBox は無料で提供されている仮想化アプリケーションである．仮想化とはホスト OS 上にソフトウェアで仮想 PC を作成して、その中で別のゲスト OS を実行するアプリケーションである．本研究ではゲスト OS として Ubuntu を導入する際に使用する．https://www.virtualbox.org/wiki/Downloads [1] にアクセスし，VirtualBox version for OS X hosts を選択し，ソフトウェアをダウンロード，GUI を用いて指示される手順の通りにインストールする．詳細は VirtualBox をダウンロードする際に UserManual もダウンロードされるのでそちらを参照してほしい

Vagrant

Vagrant は仮想環境の構築から設定までを自動的に行うことができる．つまり，本研究では VirtualBox と Ubuntu のつなぎ目の役割を果たしている．

http://www.vagrantup.com/downloads.html [2] にアクセス，MAC OS X を選択し，Vagrant をダウンロードする．インストールは mac のターミナル上で以下の通りに行う．

```
[yanase-no-MacBook-Pro:~/.vagrant.d] yanase%
vagrant plugin install vagrant-vbguest
```

Vagrant の起動, 入り方

virtualBox を起動させ, Vagrant を立ち上げるコマンドを以下に記す．これにより，phonopy コマンドを用いることが出来る．

```
[yanase-no-MacBook?-Pro:~/Vagrant] yanase% vagrant up
```

Vagrant に ssh でログインする．

```
yanase% vagrant ssh
```

UbuntuOS

UbuntuOS はコミュニティーにより開発されている Linux 系列のオペレーティングシステムであり，無償で提供されている．無償にも関わらずセキュリティーアップデートが頻繁に行われ，常に最新のパージョンを容易に手に入れることが出来る．

```
[yanase-no-MacBook-Pro:~/.vagrant.d] yanase%
vagrant box add ubuntu http://opscode-vm-bento.s3.amazonaws.com/vagrant/
virtualbox/opscode_ubuntu-13.10_chef-provisionerless.box


[yanase-no-MacBook-Pro:~/.vagrant.d] yanase% vagrant init ubuntu
```

phonopy

http://sourceforge.net/projects/phonopy [4] から phonopy をダウンロード，以下のように vagrant 上でインストールを行う．インストールは Python のパスを通すことが必須となっており，Python のライブラリは apt-get を使用する．つまり Vagrant 上での作業となるので Vagrant へのログイン方法もここで記す．

1. Python の環境を整えるために python のライブラリをインストール
2. apt-get が最新のバージョンか確認
3. ホームディレクトリに phonopy ファイルを移動
4. phonopy のパッケージを解凍
5. python の PATH を通す．このパスはそれぞれ異なるため，自分の python のファイルを探してパスを通す
6. phonopy のファイル setup.py があるディレクトリ上でインストールを行う．

```
1 sudo apt-get install python-dev python-numpy
  python-matplotlib python-tk python-lxml python-yaml
2 sudo apt-get update --fix-missing
3  mv /vagrant/phonopy-1.8.4.2-rc3.tar.gz .
4  tar xvfz phonopy-1.8.4.2-rc3.tar.gz
5  export PYTHONPATH=/usr/lib/python2.7/
6 sudo python setup.py install --home=.
```

上記のパスで python のパスが上手く通らないときは以下のように PATH を変更する必要がある．

```
# PATH の確認
echo $PATH
#パスの編集
PATH=$PATH:/home/vagrant/phonopy-1.8.4.2-rc3/lib/python
export PATH
```

setup.py が上手くいかない場合は，インストールの実行を 2 回に分けて行う．

```
sudo python setup.py build
sudo python setup.py install
```

phonopy Tutorial

1. Pre-process：モデルの構造方法
2. 力計算：VASP で得られたデータから力定数を計算する方法
3. Post-process：Phonon-DOS,FreeEnergy,Phonon-dispersion などの計算方法

Pre-process

前段階として，スーパーセルの変位は結晶の対称性を考慮し，単位結晶から作成される．例として，２×２×２構造のスーパーセルを得るために phonopy を起動する

```
phonopy -d --dim="2 2 2"
```

結果として，以下のようなファイルが生成される

```
% ls
disp.yaml  POSCAR  POSCAR-001  POSCAR-002  POSCAR-003  SPOSCAR
```

SPOSCAR は完全なスーパーセル構造であり，disp.yaml には変位に関する情報が含まれている．また，POSCAR-(001,002,003) は atom の変位のスーパーセルである.POSCAR-number は disp.yaml に書かれているそれぞれ異なった atom の変位と一致する．

力計算

Force constants（力定数）は構造のファイルである POSCAR-(001,002,003) を用いて計算する．VASP の場合，有限変位法の計算は，単に VASP 計算の POSCAR として POSCAR-(001,002,003) を使用することができる 以下が VASP の計算で用いた INCAR の例である

```
PREC = Accurate
 IBRION = -1
  ENCUT = 500
  EDIFF = 1.0e-08
 ISMEAR = 0; SIGMA = 0.01
  IALGO = 38
  LREAL = .FALSE.
ADDGRID = .TRUE.
  LWAVE = .FALSE.
 LCHARG = .FALSE.
```

構造を緩和（リラックス）させないように注意すること．その後，VASP インターフェースを使って FORCE-SETS ファイルを作成する． 微小変位したファイルそれぞれを vasp にかけること．disp-00... とは POSCAR-number と対応している．つまりそれぞれの vasprun ファイルを一緒に計算にかける

```
% phonopy -f disp-001/vasprun.xml disp-002/
vasprun.xml disp-003/vasprun.xml
```

Post-process mesh.conf

phonon-DOS, 自由エネルギーの計算に必要な mesh.conf ファイルを emacs を用いて作成する

```
ATOM_NAME = Al
DIM = 2 2 3
MP = 8 8 8
```

Post-process phonon-DOS 計算

POSCAR,FORCE-SET,mesh.conf ファイルを用意し，以下のコマンドを実行する．計算のグラフは出力できないので値を mac 上の gnuplot で表示する．

```
phonopy -p mesh.conf
```

Post-process FreeEnergy 計算

POSCAR,FORCE-SET,mesh.conf ファイルを用意し，以下のコマンドを実行すると自由エネルギーが算出される．F[KJ/mol] が自由エネルギー，S[J/K/mol] がエントロピーである

```
vagrant:~/phonopy-1.8.4.2-rc3/Al/Al-test$ phonopy -t -p mesh.conf
   __ | |__   ___  _ __   ___   _ __  _   _
 | '_ \| '_ \ / _ \| '_ \ / _ \ | '_ \| | | |
 | |_) | | | | (_) | | | | (_) || |_) | |_| |
 | .__/|_| |_|\___/|_| |_|\___(_) .__/ \__, |
 |_|                            |_|    |___/
                                     1.8.4.2-rc3
Mesh sampling mode
Settings:
  Sampling mesh:  [8 8 8]
  Supercell:  [2 2 2]
Spacegroup:  Fm-3m (225)
Calculating force constants...
Number of irreducible q-points:  20
Calculating thermal properties...
#  T [K]      F [kJ/mol]    S [J/K/mol]  C_v [J/K/mol]     E [kJ/mol]
 0.000      14.0264543      0.0000000      0.0000000     14.0264543
 10.000      14.0262949      0.0701190      0.2109276     14.0269961
      .....
```

Quasi-harmonic プログラム

結合エネルギー箇所

```
restart;with(plots):with(plottools):with(stats):with(LinearAlgebra):
with(linalg):with(ListTools):with(combinat,permute):with(Statistics):
with(StringTools):with(plots):with(LinearAlgebra):with(stats):

# E-V Curve の作成
# 結合エネルギーはユニットセルの結合エネルギーを２×２×２倍した値
p1:=[[0.95,-116.341475],[0.98,-119.249719],[0.99,-119.636394],
[1.00,-119.751101],[1.01,-119.658924],[1.02,-119.361763],
[1.03,-118.880567],[1.04,-118.234993],[1.05,-117.449962],
[1.06,-116.543482],[1.07,-115.526365],[1.08,-114.416418]];

# eV を Ry に単位変換　（8 は SuperCell を UnitCell に換算）
for i from 1 to nops(p1) do
p1[i][2]:=p1[i][2]*(2/27.2)/8;
end do;

# 各軸比× Al 格子定数=原子が体積膨張した時の格子定数
for i from 1 to nops(p1) do
p1[i][1]:=p1[i][1]*8.0986795425999993;
end do;

# 格子定数の単位をÅから a.u. に変換
for i from 1 to nops(p1) do
p1[i][1]:=p1[i][1]/0.529177;
end do;
```

```
# data11=[[各格子定数 (a.u.)],[結合エネルギー (Ry)]]
# Al は fcc 構造なので原子間距離は格子定数の sqrt(2)/2 倍である
data11:=convert(transpose(convert(p1,array)),listlist);
data1:=[(data11[1]*evalf((sqrt(2)/2))/2),data11[2]];
q1:=convert(transpose(convert(data1,array)),listlist);

# 上記の結合エネルギーの点に関数を fitting, プロットする
fit1:=fit[leastsquare[[x,y], y=a+b*x+c*x^2+d*x^3+e*x^4+f*x^5]](data1);
f2:=unapply(rhs(fit1),x);
pp1:=pointplot(q1);
pp2:=plot(f2(r),r=5..6);
display(pp2,pp1,labels=["AtomicDistance[a.u.]","BindingEnergy[Ry]"],
labeldirections=[HORIZONTAL,VERTICAL]);

# Morse 型ポテンシャルの結合エネルギー
f1:=(a,b,c,d,r)->a+b*exp(-d*r)+c*exp(-2*d*r);

# 平衡原子間距離とその結合エネルギー
x0:=fsolve(diff(f2(x),x),x=5..6);
y0:=f2(x0);

# fitting した関数を微分したものに平衡原子間距離を代入している .y2 は 2 階微分
y1:=subs(x=x0,diff(f2(x),x));
y2:=subs(x=x0,diff(f2(x),x,x));
y3:=subs(x=x0,diff(f2(x),x,x,x));

# 結合エネルギーの関数 f1 に a,b,c,d,x0 の値を代入した. まだ a,b,c,d は不明
y0=f1(a,b,c,d,x0);
```

```
# 上の式を x0 まわりで微分する．y2 はその 2 階微分
subs(x=x0,diff(f1(a,b,c,d,x),x))=0;
y2=subs(x=x0,diff(f1(a,b,c,d,x),x,x));
y3=subs(x=x0,diff(f1(a,b,c,d,x),x,x,x));

# 微分から得た式を用いて方程式を解く
eqs:={y0=f1(a,b,c,d,x0),subs(x=x0,diff(f1(a,b,c,d,x),x))=0,y2=subs(x=x0,
diff(f1(a,b,c,d,x),x,x)),y3=subs(x=x0,diff(f1(a,b,c,d,x),x,x,x))};
sol1:=solve(eqs,{a,b,c,d});

# 結合エネルギーの関数 f1 に求めた a,b,c,d を代入．新たに f3 を作る
f3:=unapply(subs(sol1,f1(a,b,c,d,x)),x);
pp3:=plot(f3(r),r=5..6,color=blue);
display(pp1,pp2,pp3,labels=["AtomicDistance[a.u.]","[Ry]"],
labeldirections=[HORIZONTAL,VERTICAL]);
```

変数早見

P1: [[倍率 a/a0],[基底状態エネルギー (eV)]]

data11: [[格子定数 (a.u.)],[結合エネルギー (Ry)]]

data1: [[原子間距離 (a.u.)],[結合エネルギー (Ry)]]

q1: data1 を list 型にコンバート　　**fit1**: fitting 関数

f2: fit1 の右辺を関数定義　　**pp1,pp2**: プロット変数

f1: Morse 型ポテンシャル関数　　**x0,y0**: 平衡原子間距離, そのときのエネルギー

y0,y1,y2,y3: f1 の微分，0,1,2,3 はその微分回数

eqs: y0,y1,y2,y3 の微分を方程式としてまとめる

sol1: eqs 方程式を解く,a,b,c,d が算出される

f3: f1 に a,b,c,d を代入して関数化　　**pp3**: プロット関数

体積弾性率箇所

```
# B1 × B2=体積弾性率の式
B1:=unapply((-lambda^3*exp(-lambda*r))/(12*Pi*ln(exp(-lambda*r))),r);
B2:=unapply((b+4*c*exp(-lambda*r))-(2/ln(exp(-lambda*r)))*
(b+2*c*exp(-lambda*r)),r);

# 単位変換
B3(r):=B1(r)*B2(r)*((27.2/2)/(0.529177^3))*160.218*10;
# 関数化, プロット
B:=unapply(B3(r),r);
evalf(B(x0));r0:=x0;
plot(B(r),r=6..8.5,color=black,labels=["AtomicDistance[a.u.]",
"BulkModulus\UTF{008E}\UTF{0087}[Kbar]"],
labeldirections=[HORIZONTAL,VERTICAL]);
```

変数早見

B1: 式 2.18 の前部

B2: 式 2.18 の後部

B3: B1,B2 の単位変換　160.218: ev/Å を GPa

※その他の数字に関しては赤本の冒頭にある単位に関するノートを参照

B: B1,B2 の乗算

r0: 平衡原子間距離=x0

デバイ温度箇所

```
#原子量（Al）
M:=26.98;
#デバイ温度の式
thetaD:=unapply(41.63*(r*B(r)/M)^(1/2),r);
plot(thetaD(r),r=6..6.4,labels=["AtomicDistance[a.u.]","Temperature[K]"]
,labeldirections=[HORIZONTAL,VERTICAL]);
```

デバイ関数箇所

```
#デバイ関数の式
Debye:=unapply((3/y^3)*int(exp(x)*x^4/(exp(x)-1)^2,x=0..y),y);
#デバイ関数の虚数値を取り除き，実数値のみにする
Df:=unapply(Re(evalf(Debye(thetaD(r)/T))),r,T):
plot(Df(r,300),r=5..6,color=black,labels=["AtomicDistance[a.u.]",
"DebyeTemperature"],labeldirections=[HORIZONTAL,VERTICAL]);
```

変数早見

M: 原子量

thetaD:デバイ温度を定義した式　式 2.38 を参照

Debye: デバイ関数を定義した式　　式 2.39 を参照

Df: デバイ関数にデバイ温度を代入し，その実数値だけをとる

自由エネルギー箇所

```
#Ry: 1Ry= 2.17987 * 10^(-18) [J]
#kb:ボルツマン定数
Ry:=2.17987*10^(-18);
kb:=1.38026*10^(-23);

#自由エネルギーにデバイ関数を導入
func:=unapply((kb/Ry)*T*(Df(r,T)-3*ln(1-exp(-thetaD(r)/T)))-
(9/8)*(kb/Ry)*thetaD(r),r,T):
plot(-func(r0,x),x=0..1000);

#自由エネルギーの式を一つにまとめる
g:=(-f3(r0)+f3(r)-func(r,T))*10^3:
f:=unapply(g,r,T):

#プロット f(AtomDistance,Temperature)
p2:=plot(f(r,100),r=5.4..5.44,color=black);
p3:=plot(f(r,200),r=5.4..5.44,color=black);
p5:=plot(f(r,400),r=5.4..5.44,color=black);
p6:=plot(f(r,800),r=5.4..5.44,color=black);

#最小値を検出
#最初は微分を使っていたが，計算時間が大きかったので最小値探索に切り替え
first100:=evalf(f(5.4,100)):
i100:=5.4:
for i from 5.4 by 0.0001 to 5.44 do
second100:=evalf(f(i,100)):
if first100 > second100 then
first100:=second100:
i100:=i;
end if:end do:
first100;i100;
mi[1]:=[i100,first100];
```

```
first200:=evalf(f(5.4,200)):
i200:=5.4:
for i from 5.4 by 0.0001 to 5.44 do
second200:=evalf(f(i,200)):
if first200 > second200 then
first200:=second200;
i200:=i;
end if:end do:
first200;i200;
mi[2]:=[i200,first200];

first400:=evalf(f(5.4,400)):
i400:=5.44:
for i from 5.4 by 0.0001 to 5.44 do
second400:=evalf(f(i,400)):
if first400 > second400 then
first400:=second400;
i400:=i;
end if:
end do:
first400;
i400;
mi[3]:=[i400,first400];

first800:=evalf(f(5.4,800)):
i800:=5.4:
for i from 5.4 by 0.0001 to 5.44 do
second800:=evalf(f(i,800)):
if first800 > second800 then
first800:=second800;
i800:=i;
end if:
end do:
first800;
i800;
mi[4]:=[i800,first800];
```

```
#最小値, 体積膨張による自由エネルギーの変化を描画
po1:=pointplot(mi[1]);   po2:=pointplot(mi[2]);
po3:=pointplot(mi[3]);   po4:=pointplot(mi[4]);
display(p2,p2,p3,p5,p6,po1,po2,po3,po4,labels=["AtomicDistance[a.u.]",
"Free energy[mRy]"],labeldirections=[HORIZONTAL,VERTICAL]);

#温度変化による自由エネルギー変化の様子 単位 [mRy]
pf1:=plot(f(r0,x),x=0.1..3000,color=red);
pf3:=plot(f(r0*1.03,x),x=0.1..3000,color=green);
display(pf1,pf3,labels=["Temperature[K]","Free energy[mRy]"],
labeldirections=[HORIZONTAL,VERTICAL]);

#温度変化による自由エネルギー変化の様子 単位 [KJ/mol]
F1:=unapply(f(r0,x)/10^3*13.60583*96.4853,x,r):
F2:=unapply(f(r0*1.01,x)/10^3*13.60583*96.4853,x,r):
FE_kj1:=plot(F1(x),x=0.1..1000,color=red);
FE_kj2:=plot(F2(x),x=0.1..1000,color=blue);
display(FE_kj1,FE_kj2,labels=["Temperature[K]","Free energy[kj/mol]"],
labeldirections=[HORIZONTAL,VERTICAL]);
```

変数早見

frac:自由エネルギーの式　　式 2.44 を参照

g:自由エネルギーの式をまとめる　　式 2.40 を参照

f: g を原子間距離と温度で関数化

p2,p3,p5,p6: 原子間距離毎の自由エネルギーのプロット変数

mi: 各軸比ごとの最小値の配列

po1,po2,po3,po4: mi のポイントプロット

pf1,pf3: 温度変化による自由エネルギーの様子をプロットする変数

F1,F2: 自由エネルギーの値を mRy から KJ/mol に単位変換

参考文献

[1] OracleCorporation,Download of Virtualbox, https://www.virtualbox.org/wiki/Downloads (accessed January 9, 2015).

[2] MitchellHashimoto,Vagrant,http://www.vagrantup.com/downloads.html (accessed January 9, 2015).

[3] UbuntuJapaneseTeam,downloadofUbuntu,http://opscode-vm-bento.s3.amazonaws.com /vagrant/virtualbox/opscode-ubuntu-13.10-chef-provisionerless.box (accessed January 9, 2015).

[4] OpenSourceDevelopmentNetworkCorporation,SOUCEFORGE.jp,http://phonopy.sourceforge.net (accessed January 9, 2015).

[5] 西谷滋人著,「固体物理の基礎」(森北出版,2006).

[6] 永山優,「体積-エネルギーカーブからの熱膨張の予測」(関西学院大学 理工学部 情報科学科 2006).

[7] 清原資之,「Ti 結晶多形における Phonon 第一原理計算」(関西学院大学 理工学部 情報科学科 2014).

[8] V. L. Moruzzi, J. F. Janak and K. Schwarz, "$Calculatedthermalpropertiesofmetals$, " Phys. Rev. B, 37(1988), 790-799.

[9] Ryoka Systems lnc,MedeA,http://materialsdesign.com (accessed January 9, 2015).