*GRRM*11 User Guide Supplement 見本

長田　有人、　前田　理、　大野　公一

1　GRRM プログラム

1-1. GRRM プログラムの基本システム

GRRM プログラムは、自身は量子化学計算を行わず、外部の量子化学計算プログラムにより得られたポテンシャルエネルギー(勾配、二次微分行列)を用いて計算を行うプログラムである。したがって、以下の図1-1のように外部の量子化学計算プログラム(Gaussian等)と本プログラムを交互に実行することによって、各ジョブを実行する。

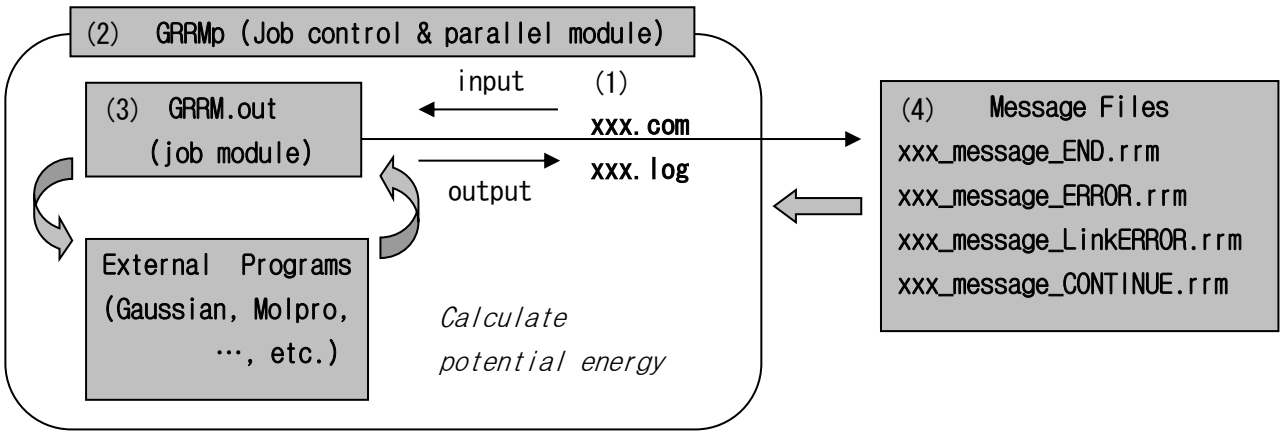

図 1-1　GRRM プログラムシステム概略図

(1) xxx.com, xxx.log (インプットファイル、アウトプットファイル)

インプットファイルは、拡張子が必ず”.com”である必要がある。拡張子が”.txt”など”.com”以外の場合には読み込みができず、エラーとなってしまうので注意が必要である。インプットファイルの書き方の詳細は 3 章で述べる。また、”xxx”はインプットファイルから拡張子(“.com”)を除いた部分(grrm.com では、”grrm”)を指し、以下**インデックス**と呼ぶことにする。GRRM プログラムの様々なファイル群はこのインデックスに従って生成する。例えば、インデックス”xxx”の場合のアウトプットファイルは xxx.log となる。ファイル群はこのようにジョブごとのインデックス名について作られるため、このインデックスを各ジョブに固有なものにしておけば、他のジョブと混乱することはない。GRRM プログラムが生成するファイル群はその他にも様々存在するが、詳細は次章で述べる。

(2) GRRMp　(実行モジュール)

GRRMp は、(3)で述べる GRRM.out と外部プログラムを交互に実行する役割を担っている。また、並列計算を行う際に、各子プロセスの処理を統合する役割がある。GRRM プログラムでジョブを行うときには、基本的に GRRMp を実行することになる。(4)で述べる Message File によって継続、終了などの命令が伝えられそれに応じてジョブを流したり、止めたりする仕組みになっている。

(3) GRRM.out (実行モジュール)

GRRM.out は外部プログラムでの計算結果を処理し、GRRM ジョブの処理を行い、計算結果を出力している。ジョブがどこまで終了したかは GRRM.out が常に把握しており、アウトプットファイル xxx.log

をはじめとした様々な出力ファイル、中間ファイルを生成する。さらに、 (4)で述べる Message Files を作ることで GRRMp に命令を伝えることによって、ジョブ全体を進行させたり、停止させたりする役割も持つ。また、GRRM.out は GRRMp から呼び出されるように設計されているため、GRRMp 経由で間接的に実行される。したがって、ユーザーが GRRM.out をコマンドラインなどで直接実行することはない。

(4) Message Files （ファイル）

GRRM プログラムのジョブの継続(CONTINUE)、正常終了(END)、異常終了(ERROR, LinkERROR)、一時停止(STOP)は、Message Files によって伝達される。xxx_message_CONTINUE.rrm, xxx_message_END.rrm, xxx_message_ERROR.rrm, xxx_message_LinkERROR.rrm は GRRM.out によって自動生成されるが、xxx_message_STOP.rrm については計算を一時的に停止する必要がある場合などに、ユーザーが必要に応じて作成する。GRRMp はこのファイルが存在するかどうかによって、判断を行っているため、内容は問題にならない。ただし、異常終了の場合のみエラーの原因が書かれているため、内容に注意が必要となる。

GRRM プログラムは、以上のようなシステムで設計がされている。同期する量子化学計算プログラムはデフォルトでは、Gaussian, Molpro となっているがユーザーが自身の環境に合わせて、選択することも可能である。

GRRMp を実行すれば、GRRM.out と外部プログラムが交互に実行され、自動的に一連のジョブが流れる。

1-2. GRRM プログラムの導入

現在リリースしている GRRM プログラムは、Linux 対応であり、Windows, MacOS 等では使うことができない。以下の導入手順は、Linux 上において行うことを前提とする。導入手順は 2 点あり、(1)実行モジュールの確認、(2)環境変数の設定　という流れで述べる。

(1) 実行モジュールの確認

実行モジュールは 1-1 でも述べたとおり、以下の 2 つである。

GRRM.out … GRRM ジョブの処理モジュール
GRRMp　　… GRRM ジョブのコントロールと並列計算を行うモジュール

2 つの実行モジュールが存在しないと正常に計算を行うことができないので注意が必要である。これらの実行モジュールは基本的には、同一のディレクトリ内に置くことが推奨されるが、置くディレクトリ名は、ユーザーが任意に決めてよく、とくに制約はない。

(2) 環境変数の設定

GRRM11 は GRRM1.22 とは異なり、環境変数の指定を行う必要がある。**ここで指定する環境変数名がプログラムの内部で呼び出されるため、この指定がなければ正常に動作しない。**環境変数は全部で以下の 4 つある。

subgrr … GRRM.out のあるディレクトリを指定するための環境変数
subgau … Gaussian プログラムのコマンド指定（通常は g03, g09）
subchk … Gaussian プログラムのユーティリティである formchk を呼び出す(通常は formchk)
submol … Molpro プログラムの実行コマンド指定（通常は"molpro -W./"）

本マニュアルでは、Linux で用いられる一般的なログインシェル(bash, tcsh)での環境変数の指定の仕方をそれぞれ示す。環境変数の設定は、/home 以下の各ユーザーのディレクトリ(~)にある、.bash_profile(.bashrc), .cshrc というファイルでそれぞれ行う。ユーザーが使用しているログインシェルによってそれぞれのファイルに以下のような行を書き加える。

bash の場合 （ .bash_profile 内に追加）

```
export GRRMroot=/usr/GRRM
export subgrr=$GRRMroot/GRRM.out
export subgau=g03
export subchk=formchk
export submol="molpro -W./"
PATH=$PATH:$GRRMroot
```

tcsh の場合(.cshrc 内に追加)

```
setenv GRRMroot /usr/GRRM
setenv subgrr $GRRMroot/GRRM.out
setenv subgau g03
setenv subchk formchk
setenv submol "molpro -W./"
set path = ( $GRRMroot $path)
```

この例は、GRRM プログラム(GRRMp, GRRM.out)を/usr/GRRM というディレクトリに置いた場合である。毎回同じディレクトリ名を記述するのが煩わしいという都合上、/usr/GRRM は GRRMroot という環境変数で置き換えている(1 行目)。2 行目以降それぞれの環境変数のデフォルトを指定しており、最後の行では/usr/GRRM にパスを通して、どこのディレクトリにいても呼び出すことができるように設定している。最低限設定が必要な環境変数は 2-5 行目の 4 つの変数であり、その他の変数、およびパスの設定などは必ずしも行わなければ動作しない訳ではない。

この設定が行われれば、Gaussian, Molpro と同期する場合は、簡単に計算を実行することができる。ただし、その他の外部プログラムと同期する場合はさらに作業が必要である。その他の外部プログラムと同期を行う場合、ユーザー自身が決まったフォーマットのファイルで読み書きを行うような翻訳プログラムを作成する必要がある。Gaussian, Molpro については、翻訳プログラムに相当する処理を GRRM.out の内部で行っているため、とくに必要がない。

以上 2 点を行えば、計算の実行についてはコマンドのみで使用することが可能となる。

３　　Ｑ＆Ａコーナー

３－１

Ｑ：ユーザーマニュアル(User Manual)に書いてある通りに、次のコマンドを投入しましたが、

GRRMp　grrm1　−p 1　−h 1

うまく動作しません。何が悪いのでしょうか？

Ａ：コマンド引数を何もつけずに、　GRRMp　　だけをコマンドとして投入したとき、

USAGE：GRRMp　[input file name]　(−p[Nproc])　(−h[Hour])

と表示されれば、プログラム本体は、正常にインストールされていて、正しく動作する状態になっています。

「USAGE：・・・」が出てこずに、「Floating exception」や「invalid ・・・」「wrong・・・・」「bad・・・」などが出る場合は、インストールされたGRRMプログラムがそのシステムに適合していないことが原因ですので、ご利用中の計算機に適合するGRRMプログラムをインストールする必要があります。

コマンドがみつからないというerrorがでるときは、(1) コマンドの打ち間違、(2)プログラムがインストールされていない、(3)プログラムへのpathが通っていない、などの理由が考えられます。

コマンドを投入したディレクトリから、GRRMpやGRRM.outを置いてあるディレクトリにpathが通っていないと、 GRRMp と打ち込んでも、正しく動作しません。マニュアルにあるように、環境設定でpathの登録をすることを推奨いたします。その他の方法でpathを設定してもかまいません。

あるいは、GRRMp や GRRM.out を置いてあるディレクトリを絶対パスもしくは相対パスで指定すれば、GRRM プログラムを起動することができます。つぎの例1）は、コマンドの投入ディレクトリ（作業ディレクトリ）の中にGRRMpとGRRM.outを置いてある場合、例2）は、ひとつ上のディレクトリに置いてある場合です。例3）は絶対パスを用いた場合です。

例1） ./GRRMp grrm1 −p1 −h1

例2） ../GRRMp grrm1 −p1 −h1

例3） /user/loca/GRRM11/GRRMp −p1 −h1

注意点1：　GRRMp がコマンド名で、ユーザーマニュアルには、このように書いてあります。ただし、各ユーザーに送付される通知書には、このコマンド名の部分を変更して使うよう指示されている場合があります。その指示に従わないと次のようなメッセージが出て、異常終了します。

GRRM11 is produced by Satoshi Maeda, Yuto Osada, Keiji Morokuma, and Koichi Ohno (C) 01−10−2011.

Use of this package (GRRM11.01) is only allowed to

**********,

who should cite the following reference when results will be published.

GRRM11.01(2011), S. Maeda, Y. Osada, K. Morokuma, and K. Ohno.

Any other person is not allowed to use this package without permission

Date of Permission: **********.

このようなメッセージがでたら、通知書を読み直し、その指示に従ってください。

なお、マニュアル通りの GRRMp のままで、GRRM を実行させたいときは、通知書の指示にしたがったコマンドを内部に書き込んだ実行用プログラムまたはシェルプログラムを作成してご利用ください。

注意点2：　上のコマンド例で、GRRM の入力ファイルであるはずの　grrm1.com　が存在しないか、その中身が

GRRM の正しい入力データでないときは、正しく動作しません。入力データのファイルの所在や名称、中身を確認してください。

注意点３： マニュアルには、コマンド引数として、-p[n] や -h[m] を必要に応じて、入力データファイル名の右に加えるように説明されています。ここで、 －pと数字の間や －hと数字の間に スペースを入れてはいけません。コマンド引数の区切りがスペース（またはタブ）になっていますので、間にスペースを入れると、別々のコマンド引数として読み込まれてしまい、正しく動作しません。

注意点４： GRRMp が正しく起動されないとき、引数をつけずに、GRRMp だけで起動を試みてるとよいでしょう。コマンド引数を何もつけずに、GRRMp を起動したとき、path が通じていてプログラムが正常に起動されれば、次のような usage がプリントされます。

USAGE: GRRMp [input file name] (-p[Nproc]) (-h[Hour])

このような表示が出てくれば、GRRMp そのものは正しく起動されています。したがって、うまく GRRMp が動作しない理由は、コマンド引数か、起動用の入力ファイルの方にあることがわかります。

上のような USAGE が出てこない場合は、GRRMp のバイナリが正しくインストールされていないか、GRRMp のバイナリが、計算環境に適合していない可能性があります。通常、GRRM11 は 64 ビット版としてリリースされますので 32 ビットの環境では正しく動きません。また、64 ビットの環境でも、その環境に適合したコンパイルがなされていないと、バイナリが正しく動作しません。お使いの計算環境で、簡単なｃープログラム（たとえば名前や日付だけをプリントするようなもの）をコンパイルしてつくったバイナリを、開発者に送り、あらためてバイナリを作ってもらうとよいでしょう。そうすると、開発者は、ユーザーの環境に適合した計算機を探し、そのような計算機が開発者側にあれば、GRRM プログラムをそのユーザー用にコンパイルし直すことができます。

３－２

Q： 利用できる計算環境では、Gaussian プログラムを呼び出すコマンド名が g09 でも g03 でもないのですが、どうしたらよいでしょうか。

A： GRRM11 のプログラム中で、Gaussian プログラムのよびだしは、subgau と subchk の2種類です。マニュアルの「インストール」のところに書かれていますように.cshrc の環境設定（B-shell の場合は１－２参照）の

setenv subgau g09

setenv subchk formchk

のところで、subgau には g09、subchk には formchk が割りつけられています。これは、通常の Gaussian プログラムの実行コマンド名が、g09 と formchk になっているからです。ご利用の計算環境で、これらの実行コマンド名が違う場合、たとえば、g09 ではなく、Gau09 ならば、

setenv subgau Gau09

としてインストールしてやれば問題ありません。formchk の方も同様です。

注意点１： Gaussian プログラムのコマンド名を、所定通りの g09 と formchk に変更できるなら、環境設定の方を変更しなくても動くようになります。

注意点２： 利用できる計算環境で、ユーザー管理下のディレクトリに Gaussian プログラムを自分でインストールできるならば、既定のコマンド名がそのまま g09 と formchk で使えるはずですので、問題なく動くはずです。

注意点３： 利用できる計算環境にある Gaussian プログラムを使うと、コマンドの引数が特別に余分に必要であったり、input ファイルや output ファイルのファイル名が、デフォルトとは違う形式であったりするときには、GRRM プログラムとのやりとりを制御するために、ユーザープログラムをつくって、コマンドを呼び出すコマンドラインや入出力ファイル名の部分を適正に変換処理すれば、動かすことは可能なはずです。setenv subgau

GauIO　として、ユーザープログラム GauIO を g09 の代わりに呼び出すようにしておき、ユーザーが作成した GauIO の中で、必要な変換処理を行うようにしておけばよいわけです。このやり方を応用すると、Gaussian プログラムの代わりに、それ以外の汎用プログラムやユーザー独自のプログラムを使って反応経路の探索を行うことも可能になります。

３－３

Q： 次のような入力データ（インプットファイル）で試してみましたが、うまく走りません。入力データ（インプットファイル）は、どのようにつくればよいのでしょうか。

```
****************************************
# OPT/B3LYP/6-31G


0 3
fe 0.000 0.000 0.000
o  0.000 0.000 1.750
****************************************
```

A： User Manual にサンプルデータが出ていますので、それを参考にしておつくり下さい。

注意点１： #で始まるルートセクションで、GRRM のオプションと計算レベルや基底関数の指定を行います。上の例では、「OPT」という指定がありますが、このようなオプションは、GRRM プログラムに存在しませんので、エラーになります。計算レベルと基底関数の指定の仕方は、Gaussian プログラムに準じていますので、上の例で問題ありません。

注意点２： #で始まるルートセクションの次の空白行は、１行だけです。ここには、ユーザーのメモを書いてもかまいませんが、行数は１行だけです。上の例では２行になっていますので、エラーになります。

注意点３： 3行目では、電荷とスピン多重度を指定します。上の例では、電荷が 0、スピン多重度が 3 ですので問題ありません。

注意点４： 4 行目以降の各行に、元素記号と3次元座標を入れます。対象とする系に含まれる原子の数だけ、入力する必要があります。元素記号は、アルファベットの大文字で始まるものです。上の例では、小文字で始まる記号になっていますので、エラーになります。

注意点５： GRRM プログラムでは、意味のある計算は3原子以上です。上の例では、原子数が２ですのでエラーになります。2 原子の場合は、2 原子間の距離に対して、1 次元的に探索するだけで簡単に調べられます。GRRM プログラムでは、超球面探索法を使っていますが、２原子の場合は、基準座標が１つしかありませんので超球面が定義できません。２原子系も扱えるようにすることも考えられますが、あまり意味があることではなさそうですので、２原子系は対象外となっています。

注意点６： 入力データには、それぞれの option ごとに、特別なデータを加える必要のある場合があります。詳細は、User Manual の説明および例をご覧ください。

３－４

Q： 反応物Rと生成物Pの間の反応経路とその上の遷移構造TSを見つけるにはどうしたらよいでしょうか？

A： 二点間中間体探索（SCW）を用いて中間体（EQ）があるかどうか調べます。中間体がないことが確認でき

たら、二点間遷移構造最適化（2PSHS）でその二点間のTSを見つけます。みつかったTSが2PSHSの両端の二点につながっているかどうかを確認するため、見つかったTSから固有反応経路追跡（IRC）を行います。

注意点1： SCWを行うときのＲとＰの座標入力の順番は、ＲとＰの間での原子間の対応関係の指定を行ったことになるので、もしも、原子の種類の違うものを対応付けると、エラーになります。原子の種類の対応付けに問題がなくても、同じ種類の原子が複数あるときに、結合の組み換えを相当複雑に行わないと対応しないような原子同士を同じ順番の位置に置いてしまうと、非常に多段階の反応でしか結ばれようがありませんので、中間体が非常に多数出てきたり、探索に非常に長時間かかったりします。遠回りの反応経路を見つけたいなら別ですが、通常は、できるだけ直結する反応経路が求められますので、ＲとＰの構造をよく調べてから、原子同士の対応付けを行う必要があります。

注意点2： SCWのRとPを逆転させると、同じ結果にならないことがあります。SCWでは、Pの構造を中心にしてRに重なる大きさの超球面から半径を縮小させながら、途中の反応経路を調べ、途中に中間体があるかどうかを調べあげます。一般に、2つの構造のうちどちらを中心に置くかで、超球面を縮小したときの反応経路（や中間体）が違ってくる可能性があります。SCWを使うときは、RとPを入れ替えて、両方試すことが推奨されます。

注意点3： SCWで中間体がみつからなければ、RとPの間が1つのTSで直接結ばれている可能性が高いので、2PSHSでその間のTS探索を行えばよいのですが、多くの場合は、RとPの間が1つのTSを介する単純な反応で結ばれているのではなく、多段階の反応で結ばれているため、中間体が出現します。Ｒ－EQ1－EQ2・・・・EQn－Pのように、n 個の EQ がＲとＰの間にみつかったときには、両端のＲ－EQ1、EQn－P、及び、隣接するEQ間について、それぞれSCWを適用して、それらの間に中間体がないかどうか調べる必要があります。この操作は、隣接する2点間すべてに SCW を適用していって、中間体がそれ以上みつからなくなるまで続けます。最初に着目したＲとＰが、著しく異なる構造の場合は、かなりの手間が必要になります。SCW で中間体がないことが確認された2点間には、2PSHS を適用し、その2点間の TS を求めます。この操作は、間に中間体がないことが確認されたすべての隣接する2点間に対して行います。念のため、TS がみつかったら IRC 追跡を適用して、2点間が TS を介して IRC でつながっていることを確かめる必要があります。

注意点4： たいていの場合、IRC 追跡を行うと、2PSHS を行った2点とのつながりが確認されますが、まれに、そうならないことがあり得ます。TS 探索法によっては、このようなこと（失敗）が結構頻繁に起きますが、2PSHS では失敗の確率はかなり低くなっています。不幸にして探索の途中で、別の反応経路にある TS が近くにあって、それがひっかかってしまうと、このような失敗が起きてしまいます。不幸にしてそうなってしまった場合は、TS から下った先の構造と、2PSHS の端の点との間で SCW をやってみるなどして、さらに反応経路のつながりを求める操作を続けてみるとよいでしょう。

注意点5： Ｒ と Ｐ の構造に、構造最適化をして求めた構造を用いるか、そうでないものを用いるかで、探索の座標系の取り方が違うので、ほんの少ししか最適化構造からずれていなくても、構造最適化した構造から行った場合と、違う探索結果が得られる可能性があります。

注意点6： ＲとＰの間に多数の中間体が存在する場合は、それらを「一筆書き」で結ぶ一連の反応経路は、一般に多数あり得ます。したがって、SCWを利用して見つかる多段反応経路は、ユニークであることはほとんどなく、また、最短であることも保証されません。

注意点7： SCW は、構造最適化された2つの構造を想定して導入された方法ですので、そうでない構造を用いた場合、とくに、いくつかの化合物もしくはフラグメントに解離した構造に適用すると、効率的な探索にならない可能性があります。