

## 物理実験 2012 年度:分子動力学

### 1 実験授業の予定

- 1 回目 説明 (テキスト、解説プリント)、アカウント、GNUPLOT, XCrysDen, 印刷
- 2 回目 説明 (テキスト、解説プリント)、計算コード、プログラムの使用法 (コンパイル、入力データファイル、計算実行、出力結果) など
- 3 回目 説明 (テキスト、解説プリント)、クラスタの安定構造と凝集エネルギー
- 4 回目 説明 (テキスト、解説プリント)、力学的保存量
- 5 回目 説明 (テキスト、解説プリント)、振動数解析プログラムの作成
- 6 回目 説明なし、振動数解析、レポートの作成

#### 実験のサポートページ

<http://f-ishii.w3.kanazawa-u.ac.jp/ja/index.cgi>

(短縮 url: <http://goo.gl/GXp62>)

の左側のメニューの「物理実験」.

### 2 計算環境の準備 (第 1 回目)

- (1) アカウントとパスワードを配布してもらう。
- (2) Linux のログイン画面がでているか確認する。WindowsNT の画面の場合は、WindowsNT 画面の指示にしたがい、シャットダウン → 再起動の手続きを行なう。再起動中に Windows または Linux の選択画面が現れるので、Linux が立ち上がるように選択する。
- (3) ログインする。
- (4) コマンドラインからの入力用の XWindow を立ち上げる。(画面の下部にあるモニターアイコンをクリックする。)
- (5) 実験に用いる計算コードを自分のアカウントにコピーする。

```
%> cp -rf ~/../cp101/buturi.jikken .
```

(6) コピーが行なわれたかどうか確認する。

```
%> ls
```

(7) 作業ディレクトリーへ移動する。

```
%> cd buturi.jikken/md
```

(8) カレントディレクトリーを確認する。

```
%> pwd
```

(9) その他の日常的に使用するコマンドを確認する。

カレントディレクトリーに存在するファイル名を知る。

```
%> ls
```

ファイル名を変える。ファイル名 filename1 をファイル名 filename2 に変えるには mv を用いる。

```
%> mv filename1 filename2
```

テキストファイルを編集するには、vi,emacs 等のエディタがよく使われる。

```
%> emacs filename &
```

や

```
%> vi filename
```

でファイル filename を編集することができる。エディタの具体的な使い方を未習得の学生は、各自 Unix や Linux などの解説本を購入し、それを参考にする。

```
%> mkdir filename
```

```
%> ls -F
```

とすると%> filename/

のように後ろにスラッシュのついた空のディレクトリが作成される。

```
%> mv filename filename2
```

とすると、ディレクトリの名前を変更することができる。

```
%> rmdir filename2
```

とすると、ディレクトリが削除される。ただし、中にファイルがある場合は消去されない。これらの他、解説本等で Unix, Linux について最低限のコマンドを授業時間中に習得すること。

### 3 ディレクトリーのツリー (tree) 構造

ファイルとはデータやプログラムを格納している箱といえる。ファイルを整理して、まとめて入れておくことのできる箱のことをディレクトリという。図 2 をみれば、いろいろな箱 (ファイル) が一まとめになっている様子が見える。

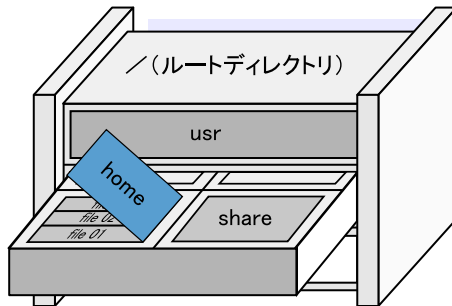


図 1: ディレクトリの概念図

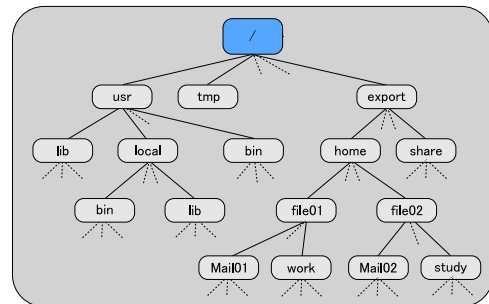


図 2: ディレクトリの階層構造

このように、ファイルはディレクトリに管理され、ディレクトリはさらに大きなディレクトリによって管理されて、次々と階層的にディレクトリがディレクトリを管理している。これを、ディレクトリツリーと言い、ファイルシステムが階層構造をした形をしている。図 2 を見ると、ファイルシステムが木のような構造をしているのがわかる。一番上のディレクトリが根元になる。このディレクトリをルート (ルートディレクトリ) と呼ぶ。

#### 3.1 カレントディレクトリとホームディレクトリ

UNIX では、ユーザごとに基準となるディレクトリが与えられ、ユーザは通常その中でのみ自由にファイルやディレクトリを作ることができる。このディレクトリをホームディレクトリと呼ぶ。また、ユーザが現在作業をしているディレクトリのことをカレントディレクトリと呼ぶ。

#### 3.2 絶対パスと相対パス

UNIX のファイルシステムでは、「/」を基準として、一意にファイルを指定することができる。この方法を絶対パス指定という。図 2 では `work` というディレクトリは、

`/export/home/file01/work`

となる。また、ファイルをカレントディレクトリからの相対的な位置によって指定することもできる。この方法を相対パス指定と呼ぶ。例えば、図 2 において `/export/home/file01/work`

がカレントディレクトリの場合、Mail01 の相対パスは、

```
../Mail01
```

となる。尚、ここでの「..」はカレントディレクトリのひとつ上のディレクトリを表している。

## 4 グラフの作成

計算結果の数値データだけから、物理的なあるいは化学的な結論や考察を引き出すことは、直観的な解釈が困難な場合が多く、解析の非効率化をもたらしてしまう。通常の物理実験と同様に、結果をすぐさまグラフにしたり、図示したり、動画化したりすることが重要である。

到達確認事項：一つの図に、2 つ以上の曲線を描き、その図を印刷することができるようになること

### 4.1 GNU プロットを用いたグラフの作成

- (1) X Window のコマンドラインから `gnuplot` を立ち上げる。(このソフトは、フリーソフトウェアで Linux や UNIX の OS では、簡便で最もポピュラーなグラフ作成ソフトの一つである。)

```
%> gnuplot
```

- (2) `gnuplot` 専用のプロンプト (コマンドを打ち込む場所) が現われる。このプロンプトに GNU プロット専用のコマンドを打ち込むとグラフが画面上の別の X Window に現われたり、描画用のファイルがファイルに書き込まれたりする。
- (3) 基本は、`plot`(2 次元の x-y 型のプロット) または、`splot`(3 次元の xyz 型のプロット) を用いて、グラフを描く。ここでは、`plot` の場合のみ説明する。GNU プロットの X Window とは違う X Window を立ち上げ、作業ディレクトリへ移動する。そのプロンプトから `emacs` 等の編集ソフトを用いて、データファイルを作成する。データは、

```
#----- データの始まり
0.1  5.0
0.2  6.0
0.3  7.0
#-----データのの終り
```

といった具合 (ただし、#の行はコメント行であり、この行がなくても正常に動作する。) にデータを用意し、テキストファイルに保存しておく。ここでは保存名を、

data.xy

すると

```
gnuplot> plot 'data.xy'
```

と、通常、データ点 (0.1,5.0)(0.2,6.0)(0.3,7.0) にプロットしたグラフが現われるはずである。

- (4) ここで現われたグラフは、データ点の描画方法やその他多くの点で各自が描きたいものと異なっているはずである。描画方法を変更するには、通常、set コマンドを用いる。例えば、データ点を直線で結びたい場合は、

```
gnuplot> set style data line
```

として、再度

```
gnuplot> plot 'data.xy'
```

または、

```
gnuplot> replot
```

とする。このように、set コマンドを用いていろいろな描画方法の指定が可能である。

```
gnuplot> help
```

として必要な言葉を入力して行けば、コマンドの解説や入力例などを探し当てる

ことができる。(ただし、解説は英語のことが多いが、ひるまずに読むようにすれば、すぐに、自分が望む入力例に近いものが見つかるはずである。どのようなグラフソフトでも、有用なグラフを作成するにはある程度の熟練が必要である。自分が描きたいグラフの形式を決めて、それが描けるように努力すればよい。特に物理学を志すまたは物理学を修めるものとして、縦軸横軸の単位を必ず示し、そのスケールの取り方にも気を配るようにする。) 以下に、役に立つ入力例を紹介する。

(5) ファイルデータが

```
#----- データの始まり
0.1  5.0  -4.0
0.2  6.0   0.0
0.3  7.0   4.0
#----- データの終り
```

といった具合(ただし、#の行はコメント行であり、この行がなくても正常に動作する。)に3列ある場合で、1列目と3列目のデータを使ってグラフを描くときは、

```
gnuplot> plot 'data.xy' using 1:3
```

とすればよい。1列目と2列目のデータと1列目と3列目のデータを用いて、2組のグラフを同じグラフ上に描きたいときは、

```
gnuplot> plot 'data.xy' using 1:2,'data.xy' using 1:3
```

とする。これらから予想されるように、異なった2つのファイルからデータを読んで、1つのグラフ上に描くことも可能である。

```
gnuplot> plot 'data1.xy' using 1:2,'data2.xy' using 1:3
```

(6) 画面上に一度描いたグラフをもう一度容易に画面上に描きたい場合が往々にしてある。そのような場合は、save コマンドで

```
gnuplot> save 'data.gpl'
```

などとしてこれまで打ち込んだコマンドを保存しておく。保存名は、任意である

がデータファイルを同名にしてはデータに上書きされてしまうので注意する。ここで、保存した `data.gpl` の中身を見ておくと大変教育的である。このファイルには、ユーザ自身が打ち込んだコマンドだけではなく、あらかじめソフトウェアが選択している環境 (各種コマンドの既定値) も書き込まれている。したがって、このファイルを直接書き換えることによりより良いグラフを作っていくことができる。また、複雑な同種の図を何枚も作る場合でも、一旦、このようなファイルを作ってしまうと、入力データを換えるだけとかいった簡単な操作で作業を遂行することが可能である。save コマンドから作られるコマンドファイル (gpl ファイル) を使うことを勧める。

- (7) GNU プロットを一旦終了 (quit) した後に、再度以前に描いた図を描きたいときは、

```
gnuplot> load 'data.gpl'
```

とする。以前に save コマンドで作ったコマンドファイルをロードすることで画面上または、ファイルに描画データが出力される。ただし、このファイルはコマンドの集まりなので、描画すべきデータの名前が変わったり、データが消去されては、再描画することはできない。

- (8) 描画グラフを印刷したり、Latex などに取り込んだりする場合は、画面上への出力を行わず、一旦ファイルに描画データを保存する。描画データのファイル形式には、通常、いろいろな形式がある。描画データをさらにどのソフトで利用するかによってその保存形式を考慮しなければならない。ここでは、eps ファイル形式で保存する方法を紹介する。プロットデータを画面上に plot または、load した後に、

```
gnuplot> set term post 20
```

などとする。このコマンドは、描画の出力 (terminal) をポストスクリプト形式 (eps 形式) で出力することを設定している。20 は、描画データ上での文字サイズ (point) を示す数で、Latex に取り込む場合や多くのひとの前でグラフを見せるときは、このくらいかまたはさらに大きいサイズがよい。

```
gnuplot> set output 'data.eps'
```

これは、出力先のファイル名を指定している。このコマンドが発行されるとここで指定されたファイル名のデータが存在した場合、ファイルの内容を消去してしまうのでファイル名の指定には、注意する。最後に

```
gnuplot> replot
```



とすればよい。data.eps を UNIX のコマンドラインの gs コマンドでみることができる。ファイルマネージャによっては、クリックするだけで eps を画面上に描いてくれるものもある。eps ファイルは、テキストファイルで書かれているので通常のテキストエディタでファイルの中身を見ることができる。この eps ファイルは、ポストスクリプト言語で書かれているので、ファイルを直接編集することも可能である。文字の大きさを変えたいなどちょっとした変更には、直接編集することもたいへん便利である。

eps ファイルを印刷する手順としては、まず gnuplot 環境を終了してから、印刷する画面を確認する。つまり

```
gnuplot> quit
```

```
%> gs data.eps
```

とする。gs コマンドは、quit で終了することができる。つぎに、

```
%> lpr data.eps
```

とすると既定の ps プリンタにデータが送られ、印刷が実行されるはずである。印刷に失敗しないように、あらかじめ印刷する前に画面上で、印刷されるデータに誤りがないかを確認した上で印刷を実行するようにする。また、紙に出しても見ないようなデータや把握することがほとんどできないような数値だけの大量なデータを印刷したりしないようにして、印刷紙の浪費をしないように心がけること。

## 5 原子構造ソフトウェア XCrySDen

- (1) 原子配置やその運動を画面上に表示してくれるソフトウェア XCrySDen の簡単な解説をする。XCrySDen がインストールされている計算機のコマンドラインから

```
%> xcrysdn &
```

で開始することができる。

- (2) 主な機能は、原子配置や原子間の結合を表示したりすることである。また、動画させる機能もあり、あらかじめ与えられたデータから原子運動の動画をみることが可能である。また、原子間距離などを測定してくれる機能や画面の図をファイル (ポストスクリプト形式) に保存することもできる。
- (3) 原子配置の入力データの作成。Xcrysden とは別に、テキストファイル編集ソフトと立ち上げて、インストールを確認するためのサンプルファイルを作成する。最も簡単な入力形式として、次のような形で、入力する。(その他にもいろいろ入力形式がある。)

```
2                <----- 原子数
comment space    <----- コメントを書くことができる。
C  0.0 0.0 0.0    <----- 1 個目の原子座標
C  1.5 0.0 0.0    <----- 2 個目の原子座標
```

座標の単位は、Å で書く。コメント欄は、空白でもよいが行をなくすことは、できない。これは、xyz 形式と呼ばれている。C の記号は、炭素原子を表しており、入力データとして必要である。入力したファイルを適当な名前前で保存する。ただし、保存したディレクトリを憶えて置くようにし、ファイル名は、oooo.xyz といった具合に拡張子.xyz を付けて置くと後で便利な場合が多い。

- (4) XCrySDen のウィンドウのメニューバーから File を選択して、上記の入力データを読み込むようにする。xyz 形式の項目を選択してファイルが保存されているディレクトリとファイル名を指定する。各種いろいろな機能があるので、メニューバーの所からいろいろ試してみるとよい。例えば、マウスを画面の原子に持って行き、動かすことにより、分子をいろいろな方向から観察することができたり、原子間の距離、結合角、二面角といった情報を取り出すことができる。実際に操作することにより、いろいろなことがすぐに分かるのでここでは、説明を割愛する。
- (5) 動画を画面上で見たいときは、上記のような 1 画面のデータを複数個連続して、1 つのファイルに書いておけばよい。

- (6) 今回は、分子動力学のフォートランのコード `tbmd.f` 中で、原子位置のデータをファイルに書き出すようにすればよい。ただし、出力データ量が多くならないように 5 ステップ毎または 10 ステップ毎に原子位置のデータを出力する。
- (7) あらかじめ用意されているサンプルファイルもあるので、メニューバーから File を選択して、XCrysDen Examples からファイルを選択して描画された構造を眺めるておくとよい。

## 6 プログラミングに慣れるための演習

下記のサイトにある PDF ファイルをダウンロードして、その指示に従って演習を進める。与えられてプログラムコードを入力し、コンパイル、入力データ設定、実行、結果の図示を行う。うまく行けばこの演習は終了。

<http://cphys.s.kanazawa-u.ac.jp/~ishii/jikken-exercise.pdf>

## 7 分子動力学計算コード (第2回目)

### 7.1 ソースコードのコンパイル (実行形式ファイルの作成)

(1) メインプログラムのソースコードは、

```
tbmd.f
```

である。ソースコードは、全てフォートラン (Fortran 言語) で記述されている。メインプログラムでは、階層的に副プログラムが多数使われており、メインプログラムと同様に `xxxx.f` のファイル名形式にする習慣になっている。

(2) これらの複数のソースコードから、実行形式のプログラムを生成する。生成過程には、コンパイルとリンクと呼ばれる作業がある。しかし、通常は、`f77` などのコンパイルコマンドを用いて一連の作業を行うようになっている。これらの作業は、プログラム開発においては、毎回一定の手続きなので、`make` を用いて毎回の作業工程を簡便にする方法を用いるが常套的なコンパイル方法となっている。単なるコンパイル作業は、`xxxx.f` ファイルからオブジェクトファイルと呼ばれるファイル `xxxx.o` を作成する。単なるリンク作業は、必要なオブジェクトファイルを結びつけて、実行形式のファイルを作成する。

(3) 基本的に、

`make` は、`Makefile`(または、`makefile`,`MAKEFILE`) に記述されている一連の作業工程にそってコマンドを実行してゆくツールである。

このようなツールには、各種あるが、`make` のもっとも重要な特徴は、こちらで指定したファイルが書き換えられた日付を基に、スキップできるはずの作業工程を行わないことである。ここでは、複数のソースコードを使用しているが、`make` を使用することにより、すでに正常にコンパイルを終えたソースコードについては、作業を省略することができる。つまり、`make` は、プログラム開発の過程で、その作業時に修正したファイルのみを、再コンパイルするなどの、必要最小限の作業を自動的に行うことを可能にする便利なコマンドである。将来、プログラム開発に携わる人には、必須のコマンドである。しかし一般ユーザもある程度の使い方知っておくことで格段に作業を早めることができる。

(4) `makefile` に記述されている実行形式のプログラム名は、

```
tbmd.out
```

である。`make` コマンドを実行する際には、ここでは、

```
%> make tbmd.out
```

と打ち込むことにより、実行形式ファイルが生成されるように設定されている。

- (5) 初歩的な makefile の記述形式は、難しくないので、makefile を読んでみて、見よう見まねで覚えることをお薦めする。
- (6) ここまでをまとめると、ソースコードを作成したら (または、コードを修正したら)、make コマンドにより、実行形式ファイルの再作成を行えばよいということである。

## 7.2 入力データの作成

- (1) 与えられた計算コードでは、入力ファイルに対して `yyyy.dat` 名前を当てている。ほとんどの入力ファイルが計算には、必要であるが、特に、重要なのは、

`ccluster.dat`,

`md_str.dat`,

`md_tb.dat`,

`mdinput.dat`,

である。実験課題を行うのにここで挙げた 4 つのファイルの全てを熟知する必要はない。それぞれのファイルに数箇所だけ重要な入力データがあり、それを覚えておけばよい。

`mdinput.dat` で原子数、時刻の刻幅、計算する総ステップ数を与え、

`ccluster.dat` に原子の初期座標を与える

ようになっている。

- (2) 最初に与えたファイル群をそのまま用いることにより、炭素原子 2 個の集合体の分子動力学計算ができることになっている。このときの様子や計算するときの手続きをよく覚えておくと、原子数を変えた場合の計算で不具合が発生しても、解消の糸口を掴めるかもしれない。

**重要:** はじめて計算する場合は、  
`mdinput.dat`  
の 43 行目の左端の数字 2 を 0 に変更すること。

2 MD\_init (0,1)

を

0 MD\_init (0,1)

に変更すること。この

`MD_init`

というパラメータは

- 0: ccluster.dat の座標を読み込む
- 1: 前回の計算結果の座標からリスタートする
- 2: 前回の計算結果からリスタートする (速度も読み込む)

という意味をもっている。他にも多くのパラメータがあるが、多くがソースファイルにその記述があるので、自分で調べてみるとプログラムがよく理解できて楽しいかと思う。

### 7.3 計算実行

- (1) どの様に動作するか分からないコマンドをむやみにやたらに実行するべきではない。(ひょっとしたらそのコマンドは全てのファイルを強制的に消去するコマンドかもしれない。) しかし、今回は、素早くプログラムを理解するために、

```
%> ./tbmd.out
```

を実行してみる。実行すると出力情報の一部がウインドウ上を流れてゆく。計算が終わってしまうと最初に流れていった情報は、失われてしまっている。そこで、下記のように画面にでる情報をファイルに落すようにする。

```
%> ./tbmd.out >& log &
```

最後の記号&は、必要ないが、計算をバックグラウンドで処理してくれるので、長い計算の場合、計算している間に、他の作業 (レポートの草案などを書く作業) をすることができる。

- (2) 計算実行後、出力したファイルの一つである log を開いて、実行結果を確認する。分子動力学の各ステップの各種エネルギーが出力されているはずである。log ファイルの最後の方には、最終的な原子座標と原子に働く力が出力されている。力が小さくなりステップ回数を増やしても原子の座標が変化せず、エネルギー値も収束してゆけば、安定構造に収束したと推察される。系の摩擦をゼロ (mdinput.dat 中で  $eatamd = 0.0$ ) とした場合は、原子は、運動し続けるはずである。
- (3) 計算データを解析して、安定な原子集合体の幾何学的形やその時のポテンシャルエネルギーを求める。また、凝集エネルギーについても算出し、考察を加えるなどの作業をする。2 原子の場合は、ポテンシャル曲線から容易に振動数を計算することができるはずである。原子数が多くなるといろいろな準安定構造が見つかるかもしれないので初期配置などを予想したり、古典力学や量子力学または電

磁気学の知識や経験に基づいて配置を工夫することにより、何度も計算実験を行いエネルギー的に安定な構造を模索する。

- (4) 計算機実験の実習要領を参考に、レポートを作成する。独自のアイディアで、実験を行なうのもよいが、大容量のファイルを書き出さないように注意する。

### 重要

原子数を増やしたりする場合や、安定構造を計算する場合, `mdinput.dat`

ファイルの 5 行目の左側の数字：原子数と

`ccluster.dat`

ファイルの 1 行目に原子数

2 行目以降に原子数の数だけ座標  $x,y,z$  (単位は nm)

を変更する。

## 7.4 ファイルのバックアップ等に関する注意

計算結果や入力データについて、こまめにディレクトリを作成し、計算を再現できる様に整理して保存しておくことが実験を効率的にすすめるコツとなるであろうと思われる。例えば、原子数や原子の初期座標別にディレクトリ名を付ける。

```
%> mkdir C6
%> cd C6
%> cp ../*.dat .
```

とし、`ccluster.dat`, `mdinput.dat` を編集してから、

```
%> ../tbmd.out
```

のようにするとよい。また、USB メモリ等にそれらのバックアップをとっておくと、家で結果の解析等が可能になる。間違えて全てのデータを消してしまったりした場合にも再実験が少なくすむのでこまめにバックアップをとることを推奨する。



## 7.5 原子位置の出力

- (1) ソースコードをエディタで編集して、原子位置が出力されるようにする。

tbmd.f 中の 10 の文番号の前に、下記のような WRITE 文を挿入すると、原子位置のデータが 10 ステップ毎に出力されるようになる。

```

      if( i_time_step/10*10.eq.i_time_step )then
        fac=10.
        write(16,*) N_atom
        write(16,*) 'i_step=',i_time_step
        DO 20 i_atom=1,N_atom
          write(16,100)coord0(i_atom,1)*fac,coord0(i_atom,2)*fac,
&                                     coord0(i_atom,3)*fac
100      format('C ',3(1X,E15.6))
        20  continue
      endif

```

ただし、FORTRAN77 の文法では、次のような形式を守らなければならない。上記の if 文の前には 6 カラム (半角スペース 6 個) が存在し、本文は、7 カラム目から 72 カラム目までに記述する。各行の 1 カラムは、コメント文の記号などに使われ、2 カラムから 5 カラムまでは、文番号に使用される。6 カラム目は、継続行の印を付けるために使用される。

- (2) プログラムを修正したら、再コンパイル (make) を行い、実行形式ファイル (tbmd.out) を更新してから、計算を実行する。
- (3) 正常に実行が終了すると fort.16 が生成されている。これを XcrysDen を起動して、読み込み、原子運動の様子を観察する。

## 7.6 データの出力

tbmd.f 中の適当な場所 (10 の文番号の前) に、下記のような WRITE 文を挿入すると、系の運動エネルギー、ポテンシャルエネルギー、全エネルギーがファイル番号 17 に出力される。

```
write(17,711)i_time_step,  
&          E_atom_kinetic,  
&          E_total_per_atom,  
&          E_total_system  
711 format(i8,f15.8,f15.8,f15.8)
```

全運動量を出力するには、質量と速度のデータを用いて、計算を行なった後その数値を出力する。そのとき単位を明らかにさせておくことでレポートの作成時に便利である。(下の例では、fac=1.0 としてあり単位については考慮していない。)

```
xm=0.0  
ym=0.0  
zm=0.0  
fac=1.0  
do 712 i_atom=1,N_atom  
    xm=xm+mass_atom*velo( i_atom,1 )  
    ym=ym+mass_atom*velo( i_atom,2 )  
    zm=zm+mass_atom*velo( i_atom,3 )  
712 continue  
    write(18,713)i_time_step,xm*fac,ym*fac,zm*fac  
713 format(i8,e15.6,e15.6,e15.6)
```

上記の例を参考にすれば、全角運動量の出力を作ることは容易であろう。

プログラムを修正したら、再コンパイル (make) を行い、実行形式ファイル (tbmd.out) を更新してから、計算を実行する。

## 8 振動数解析

分子内振動の振動数を求める具体的方法は、いくつか考えられるが、ここでは、簡略化されたフーリエ変換を用いる方法を実習する。(一般的な方法は、テキストで説明されているが、数値計算の量が多い)

### 8.1 フーリエ変換を用いる方法

フーリエ変換 分子動力学では、原子位置とその速度の時間変化が、時系列で得られる。つまり、

$$u_{I\alpha}(n\Delta t), \quad \dot{u}_{I\alpha}(n\Delta t), \quad \alpha = x, y, z, \quad n = 0, 1, \dots$$

が得られる。例えば、原子位置のデータから時間に対してフーリエ変換

$$A(\omega) = \int_0^\infty u_{I\alpha}(t) \cos(\omega t) dt$$

を計算し、スペクトル関数  $A(\omega)$  を得る。 $u_{I\alpha}(n\Delta t)$  が振動数  $\omega_1$  の運動を含んでいると  $A(\omega)$  の振動数  $\omega_1$  のところにピーク構造が現れる。このようにして  $A(\omega)$  を計算することにより振動数  $\omega_1$  を得ることができる。フーリエ変換に速度  $\dot{u}_{I\alpha}(n\Delta t)$  を用いる場合や  $\sin(\omega t)$  を用いる場合などいろいろ実験することができる。通常、この方法では、振動数の中心が分かりにくいので、振動数がゼロの付近にピーク構造が数値誤差として大きく現れてしまう。これを無くすには振動の中心をあらかじめ求めておき、振動の中心からの変位として原子位置  $u_{I\alpha}(n\Delta t)$  を定義しておくか、または、速度を用いてフーリエ変換を行うとよい。

実験手順としては、 $A(\omega)$  を  $\omega$  の関数として計算できるように計算コード `tbmd.f` を改良する。

$A(\omega)$  の出力データをグラフ化する。

$A(\omega)$  の数値データから、振動数を読み取る。

振動の振動数だけではなく、振動様式までもこの方法で求めることが可能であるが、ここでは、概略だけを述べる。全ての原子  $I$ 、振動方向  $\alpha$  について、スペクトル関数  $A(\omega)$  を求め、固有振動の構造が現れたピーク構造周辺だけのデータを用いて、逆フーリエ変換を行なうと、固有振動する各原子が時間とともにどのように振動しているかが分かる。

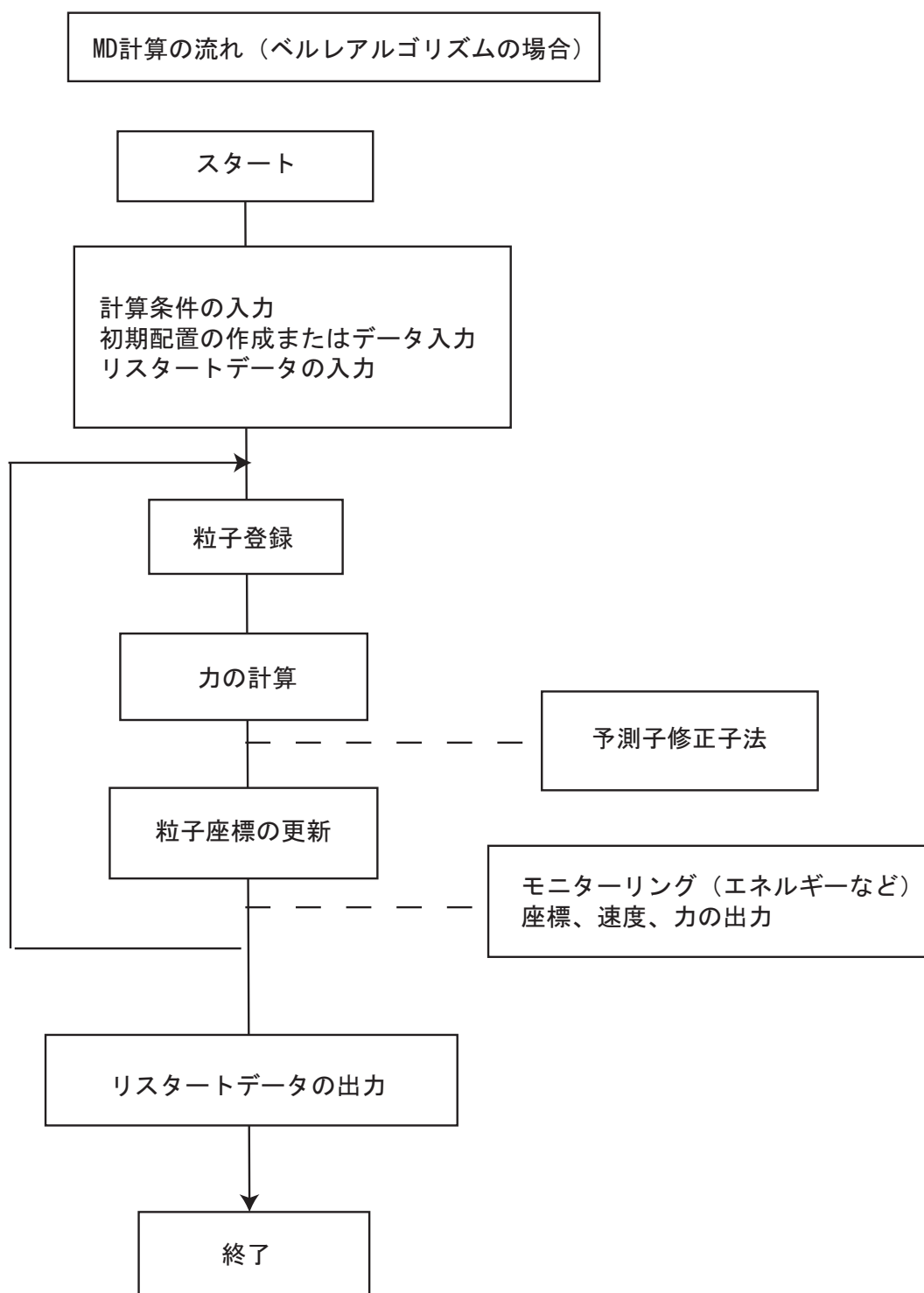


図 3: 分子動力学コード (ベルリアルゴリズム使用) のフローチャート

## 9 分子動力学計算実験実習要領

### 9.1 課題、諸注意

- (1) 実習に使うパソコンまたはワークステーションに実際にログインし、計算に使うプログラムを作製または整備し計算できる環境を構築または確認する。
- (2) 分子動力学計算を行なうプログラムが用意できたら摩擦係数を入れた分子動力学計算を炭素クラスターに適用し原子数を固定した場合にどのような配置のクラスターが安定であるか求める。その際
  - 最初の内は、分かりやすい初期配置を与え効率的に作業を進めるように注意する。
  - 初期配置により結果が変わる場合はよく考察を加えたり計算条件を変えたりしてクラスターが安定であるかどうか検討をする。重要であると考えられる場合には、提出レポートで議論をする。
  - クラスターの原子数を増やしていく。ただし 2 原子分子などの小さなクラスターである程度計算に慣れたり、計算プログラムの修正を行なうこと。
  - 原子位置の座標を xyz 形式でファイルにソフトウェア XCrySDen(次節で説明)で原子構造を考察する。
  - 安定なクラスターの形状に注目する。原子間距離や結合角など求める。なぜそのような形状を取るのかなどを考える。
  - クラスターの凝集エネルギー (クラスターを構成する原子をバラバラに引き離すのに必要なエネルギー) についての議論を行なう。原子数に対して原子当たりの凝集エネルギーを GNU プロットなどを用いて図示する。
- (3) 摩擦係数を入れないで、分子動力学計算を行ない、運動の保存量 (全エネルギー、全運動量、全角運動量) がどの程度保存しているかを GNU プロットを用いて図示したり、平均値や分散を計算したりして解析を行なう。(このためには、新たに、ソースコードに物理量の出力をする部分を書き加える必要がある。)
- (4) 安定なクラスターを用いてそのクラスターの分子内振動を分子動力学を用いて測定する。その際、
  - 対称性のよいクラスターであれば異なった対称性の振動を誘起しない振動を初期配置として与えるなどして直接分子内振動を観察し振動数を求める。
  - または、原子配置を逐次変化させながらエネルギーを求め、ポテンシャルの 2 次の係数より振動数を計算する。
  - テキストにあるように本格的に分子動力学法を用いて速度相関関数などのフーリエ成分より振動の状態密度分布を計算する。

- (5) 計算方法や計算結果をグループで議論することは大いに推奨できることであるが、各自、自由な発想の基に個人個人計算を行なうこと。