

# チュートリアル

- 1      プリオンタンパク質と低分子化合物の複合体

# 1 プリオンタンパク質と低分子化合物の複合体

*Paics View* の使い方を示す例として、プリオンタンパク質の C 末ドメインとプリオン病阻害薬である GN8 ( *Proc. Natl. Acad. Sci. USA*, 104, 11921 ) の複合体の計算を行う手順を示す。( マニュアルの「使い方」も参照のこと。)

## 1.1 複合体の構造

使用する構造は、*Paics View* と一緒に配布された、

sample\_prion\_gn8.pdb

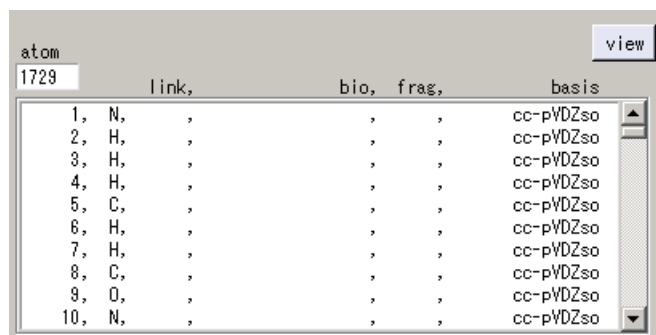
に記述されている。プリオンタンパク質の C 末ドメイン ( 残基番号 124 ~ 226 ) は、1666 原子、103 アミノ酸残基から構成されており、179 番と 214 番が SS 結合を形成している。また、GN8 は、岐阜大学人獣感染防御研究センターで発見されたプリオン病の阻害薬であり、63 原子から構成されている。従って、原子の総数は 1729 となる。添付の構造は、プリオンタンパク質の PDB 構造 ( 1AG2 ) に、GN8 を加え、古典計算を実行し得られたものである。

## 1.2 構造を読む

*Paics View* を起動し、以下の手順で構造を読む。

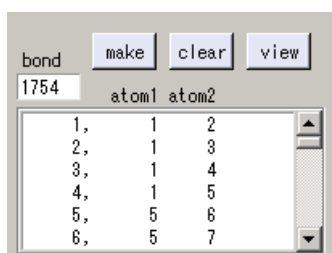
1. file メニューから read を選択し read molecule window を開く。
2. browse ボタンを押してファイル選択ウインドウを開く。
3. sample\_prion\_gn8.pdb を選択する。( 拡張子が .out のファイルも添付されているので、これと間違わないようにすること。)
4. テキストボックスにファイル名が入力され、ファイルタイプとして pdb が選択される。( ファイルの拡張子が .pdb の場合は、ファイルタイプが自動的に pdb となる )。
5. read ボタンを押し、構造を読む。
6. gui window で、1729 個の原子が読まれ ( 図 1 )、1754 個の結合が生成された ( 図 2 ) ことが確認できる。結合は原子間距離から自動的に作成される ( 詳しくは、マニュアルの「使い方」を参照 )。
7. 同時に、opengl window に、構造が描画される。

図 1: 構造を読んだ後の gui window の原子の表示



atom	link,	bio,	frag,	basis
1729				
1, N,	,	,	,	cc-pVDZso
2, H,	,	,	,	cc-pVDZso
3, H,	,	,	,	cc-pVDZso
4, H,	,	,	,	cc-pVDZso
5, C,	,	,	,	cc-pVDZso
6, H,	,	,	,	cc-pVDZso
7, H,	,	,	,	cc-pVDZso
8, C,	,	,	,	cc-pVDZso
9, O,	,	,	,	cc-pVDZso
10, N,	,	,	,	cc-pVDZso

図 2: 構造を読んだ後の gui window の結合の表示



bond	atom1	atom2
1754		
1,	1	2
2,	1	3
3,	1	4
4,	1	5
5,	5	6
6,	5	7

ここで、

- 結合の clear ボタンを押と、結合がクリアされる。
- 再び make ボタンを押すと、結合が再定義される。

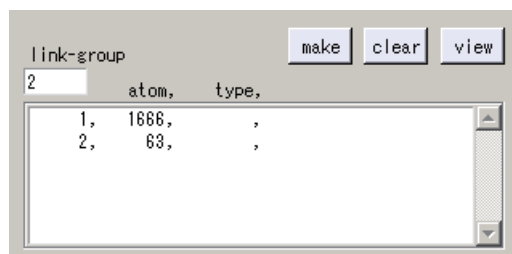
### 1.3 link-group を作成する

以下の手順で link-group を作成する。

1. gui window の link-group の make ボタンを押す。
2. link-group が 2 個作成され、gui window のリストボックスに表示される (図 3)。一方は、原子数 1666 の link-group で、もう一方は、原子数 63 の link-group である。

ここで、

図 3: make ボタンが押された後の gui window の link-group の表示



- リストボックスの link-group をダブルクリックすると、link-group window が開き、その link-group のみが描画される。
- link-group の clear ボタンを押すと、link-group がクリアされる。
- 再び make ボタンを押すと、link-group が再定義される。

#### 1.4 bio-unit を作成する

以下の手順で bio-unit を作成する。

1. gui window の bio-unit の make ボタンを押す。
2. コマンドプロンプトにアルファ炭素の情報が書き出される (図 4)。
3. bio-unit が 103 個作成され、残基の種類が自動的に識別される。また、その結果が、gui window のリストボックスに表示される (図 5)。
  - 1 番目の bio-unit は、 $-\text{NH}_3^+$  の N 末なので、N(+1) の表記が追加される。
  - 103 番目の bio-unit は、 $-\text{COO}^-$  の C 末なので、C(-1) の表記が追加される。
  - 56 番目の bio-unit は、91 番目の bio-unit と SS 結合を形成しているので、SS-91 の表記が追加される。
  - 91 番目の bio-unit は、56 番目の bio-unit と SS 結合を形成しているので、SS-56 の表記が追加される。
  - 1 番目の link-group のタイプが、PEPT と変化する。これは、bio-unit を定義することで、1 番目の link-group がペプチド鎖であると識別されたからである。
  - また、res=001 となっているのは、ペプチド鎖の最初の残基番号が 1 にセットされていることを示している。

図 4: make ボタンが押された後のコマンドプロンプトのログ

```

-----
# make bio-unit @ link-group 1
-----

number of atoms          =      1666
number of atoms ( hydrogen ) =      802

103 C-alfas were found ( ratio = 16.17 )

  1  C-alpha :      5 , C :      8 , N :      1 ,
  2  C-alpha :     12 , C :     27 , N :     10 , C-side :     14 ,    15
  3  C-alpha :     31 , C :     34 , N :     29 ,
  4  C-alpha :     38 , C :     41 , N :     36 ,
  5  C-alpha :     45 , C :     62 , N :     43 , C-side :     47 ,    17
  6  C-alpha :     66 , C :     79 , N :     64 , C-side :     68 ,    13
  7  C-alpha :     83 , C :     98 , N :     81 , C-side :     85 ,    15
  8  C-alpha :    102 , C :    105 , N :    100 ,
  9  C-alpha :    109 , C :    116 , N :    107 , C-side :    111 ,     7
 10  C-alpha :    120 , C :    126 , N :    118 , C-side :    122 ,     6
 11  C-alpha :    130 , C :    143 , N :    128 , C-side :    132 ,    13
 12  C-alpha :    147 , C :    154 , N :    145 , C-side :    149 ,     7
 13  C-alpha :    158 , C :    178 , N :    156 , C-side :    160 ,    20
 14  C-alpha :    190 , C :    192 , N :    180 , C-side :    187 ,    11
 15  C-alpha :    196 , C :    209 , N :    194 , C-side :    198 ,    13

```

4. 1 番目の link-group をダブルクリックし、link-group window を開く。
5. bio-seq. のテキストボックスで、001 124 とし、set ボタンを押す。
6. ペプチド鎖の最初の残基番号が 124 に変化する。

ここで、

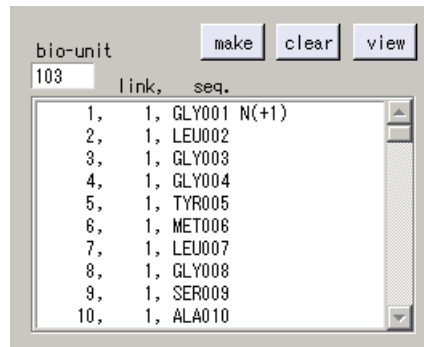
- リストボックスの bio-unit をダブルクリックすると、bio-unit window が開き、その bio-unit のみが描画される。
- bio-unit の clear ボタンを押すと、bio-unit がクリアされる。
- 再び make ボタンを押すと、bio-unit が再定義される。

## 1.5 fragment を作成する

以下の手順で fragment を作成する。

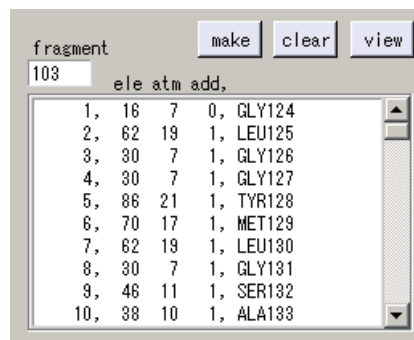
1. gui window の fragment の make ボタンを押す。
2. fragment が 103 個作成され、gui window のリストボックスが更新される ( 図 6 )。

図 5: make ボタンが押された後の gui window の bio-unit の表示



- 1 ~ 102 番の fragment は、1 番目の link-group から作られており、103 番の fragment は 2 番目の link-group から作られている。
- 1 ~ 102 番の fragment は、bio-unit に対応している。
- 56 番目の fragment は、CYS179 と CYS214 を合わせて 1 つのフラグメントとなっている (SS 結合があるため)。

図 6: make ボタンが押された後の gui window の fragment の表示



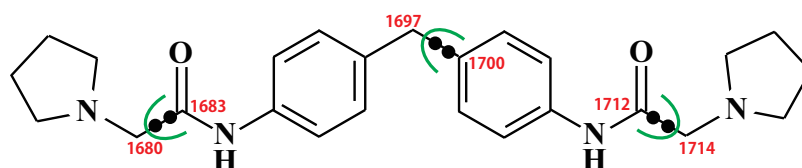
ここで、

- リストボックスの fragment をダブルクリックすると、fragment window が開き、その fragment のみが描画される。
- fragment の clear ボタンを押と、fragment がクリアされる。
- 再び make ボタンを押すと、fragment が再定義される。

## 1.6 GN8 の手動分割

この段階で、基本的な fragment の定義が出来ているが、103 番目の fragment は、GN8 (63 原子) がまるごと 1 つの fragment になっている。そこで、次は、この fragment をさらに 4 つに分割する。GN8 の分割箇所を図 7 に示す。

図 7: GN8 の分割箇所 ( 図中の数字は分割箇所の原子の通し番号 )。



手順は以下の通り。

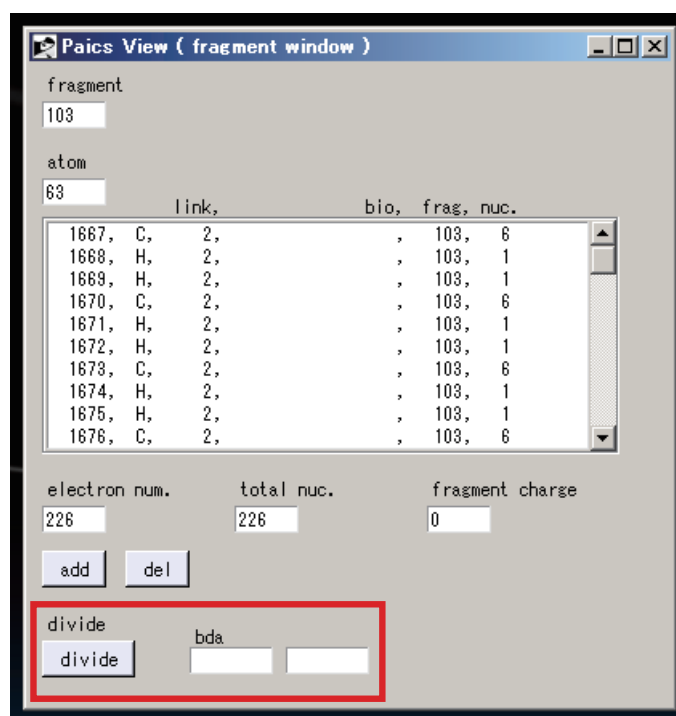
1. gui window の fragment のリストボックスの 103 番目の fragment をダブルクリックする。
2. fragment window が開き ( 図 8 )、opengl window には 103 番目のフラグメントが描画される ( f キーで拡大し、a キーで原子の番号を描画したのが図 9 )。
3. まず、1697 番と 1700 番の原子間で分割を行う。
4. fragment window の一番下のテキストボックスに、2 つの原子の番号を入力し、divide ボタンを押す。この際、bda の方に 1697 番を入力する。
5. divide ボタンを押すと、fragment window が閉じる。
6. gui window の fragment 数が 104 になり、リストボックスの 103 番目と 104 番目のフラグメントの表示が、

```
103, 110 30 1,  
104, 116 33 0,
```

となる。これは、もともと 63 原子から構成されていた 103 番目の fragment が、30 原子の fragment と 33 原子の fragment に分かれたことを意味する。

7. 次に、再び、103 番目のフラグメントをダブルクリックし、fragment window を開く。この場合、GN8 の片側半分が opengl window に描画される。

図 8: 103 番目の fragment に関する fragment window。赤で示したのが、手動分割を行う部分。



8. 1714 を bda のテキストボックスに入力し、1712 をもう一方のテキストボックスに入力し、divide ボタンを押す。
9. gui window の fragment 数が 105 になり、リストボックスの 103 ~ 105 番目の fragment の表示が

```

103,  46  16   0,
104,  64  14   2,
105, 116  33   0,

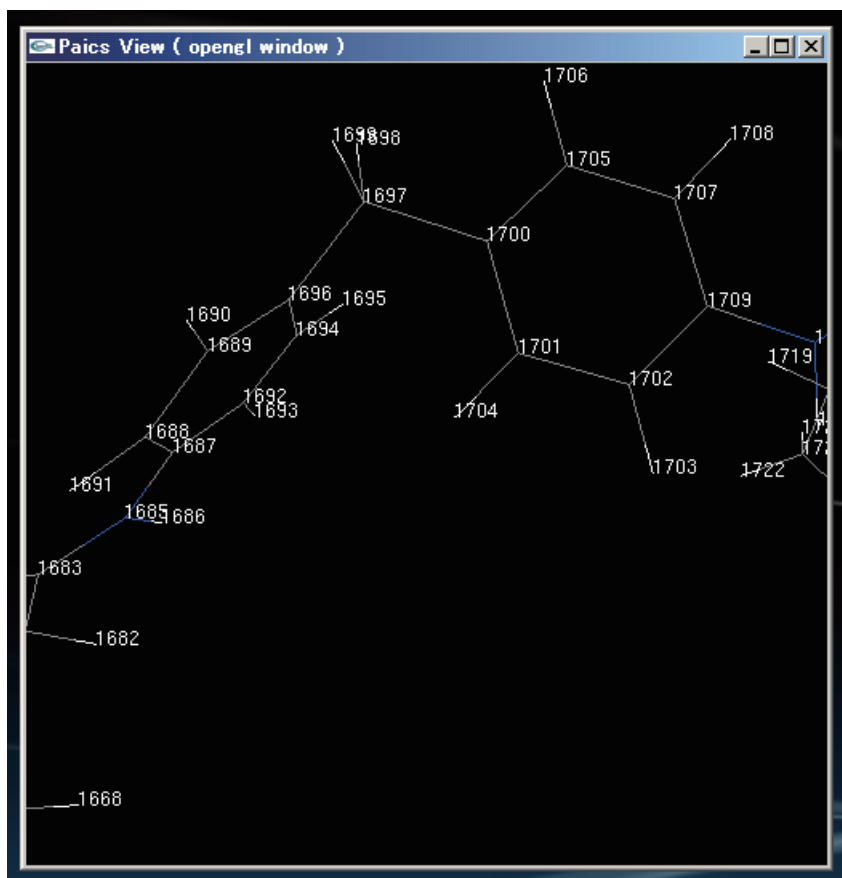
```

となる。これは、30 原子から構成されていた 103 番目の fragment が、16 原子の fragment と 14 原子の fragment に分割され、もともと 104 番目だった fragment が 105 番目にシフトしたことを意味する。

10. 最後に、105 番目のフラグメントをダブルクリックし、fragment window を開く。
11. 1680 を bda のテキストボックスに入力し、1683 をもう一方のテキストボックスに入力し、divide ボタンを押す。



図 9: 103 番目の fragment window を開き、拡大しおよび原子の番号を描画した opengl window。



12. gui window の fragment 数が 106 になり、リストボックスの 103 ~ 106 番目の fragment の表示が

```
103,  46  16  0,
104,  64  14  2,
105,  70  17  1,
106,  46  16  0,
```

となる。これは、33 原子から構成されていた 105 番目の fragment が、17 原子の fragment と 16 原子の fragment に分割されたことを意味する。

13. 以上で、GN8 が 4 つの fragment に分割された。

## 1.7 入力ファイルを書き出す

この fragment の定義の状態で、

1. file メニューから write を選択し write molecule window を開く。
2. browse ボタンを押してファイル選択ウインドウを開く。
3. ファイル名を決める（ここでは、sample\_print\_gn8.paics.inp とする）。
4. write ボタンを押し、入力ファイルを書き出す。
5. 入力ファイルが作成されていることを確認し、コマンドプロンプトに exit と打ち込み、*Paics View* を終了する。

作成された入力ファイルの最初の部分は、

```
mpi_np 1
mem_mbyte 1792

ATOM
      1729      1
      1      7      cc-pVDZso_007      43.057410      76.048248      80.942639
      2      1      cc-pVDZso_001      44.179907      75.949983      82.482766
      3      1      cc-pVDZso_001      43.216147      74.423084      79.954312
      4      1      cc-pVDZso_001      41.247052      76.295803      81.494439
      .
      .
      .
```

となっている（各キーワードの意味は、*PAICS* のマニュアルを参照）。このファイルに必要なキーワードを加えて、目的の計算を実行する。

[ 注意 1 ]

入力ファイルを作成する際、link-group や bio-unit は使われず、fragment の定義のみ参照される。link-group や bio-unit は、あくまで、fragment を定義する過程で必要なので定義される。

## 1.8 計算を実行する

ここでは、上で作成した入力ファイルに、

```
ri_cmp2_chk 1
```

の1行みを追加し、計算を実行する。こうすることで、HF計算に加えて、RI-MP2計算が行われる。mem\_mbyte は、1CPU (コア) 当たりのメモリを Mbyte 単位で示しており、必要ならばこの値を下げて実行すること (とりあえず、テスト計算なので、値を大きくする必要は無い)。mpi\_np はこのままでよい (使用する CPU 数を指定しているわけではない)。実行のしかたは、*PAICS* のマニュアルを参照すること。参考として、8 コア (XeonE5429) 使用した場合、約 38 時間掛かった。*Paics View* の配布と一緒に、計算済みの出力ファイル

```
sample_pri on_gn8. paics. out
```

を添付するので、これを用いて、以下のチュートリアルを進めてもよい。

## 1.9 計算結果を読む

計算結果が得られたら、*Paics View* を起動し、出力ファイルを開く。手順は以下の通り。

1. file メニューから read を選択し read molecule window を開く。
2. browse ボタンを押してファイル選択ウインドウを開き、出力ファイルを選択する。
3. ファイルタイプとして paics-out を選択する。(ファイルの拡張子が「.out」であれば、自動的に paics-out が選択される。)
4. read ボタンを押す。
5. gui window で、原子が 1729 個、結合が 1754 個、fragment が 106 個作成されたことを確認する (出力ファイルに書き出されているのは、計算を行った時の fragment の定義なので、この段階で、既に GN8 は 4 分割されている)。link-group と bio-unit の情報は出力ファイルには記述されていないので、この段階では定義されない。

## 1.10 link-group と bio-unit を作る

fragment が定義されていれば、*Paics View* を使って相互作用エネルギーの解析を行うことが出来るが、fragment とアミノ酸残基の対応を見るために、bio-unit を定義しておいた方がよい。

1. link-group の make ボタンを押して 2 個の link-group を作る。

2. bio-unit の make ボタンを押して 103 個の bio-unit を作る。この段階で、fragment のリストボックスに bio-unit の情報が追加される。(しかし、残基番号が 1 番からはじまっている。)
3. link-group のリストボックスで、1 番目の link-group をダブルクリックし、残基番号の始まりを 124 番にする。

[ 注意 1 ]

ここで、fragment の make ボタンを押さないこと。理由は以下の通り。fragment が定義されている状態で make ボタンを押すと、一度定義がクリアされ、再度自動的に fragment が定義される。この際、入力ファイルの作成で行ったように、GN8 は 1 つの fragment となり、fragment の総数は 103 個となる。計算結果は、106 個の fragment に対して実行されたものなので、計算結果と fragment の定義に矛盾が生じる。このような矛盾が発生した場合、計算結果を破棄する仕様となっている。

### 1.11 相互作用を解析する

ここでは、プリオンタンパク質と GN8 の相互作用を、*PaicsView* の機能を使って解析する。(もちろん、出力ファイルには、フラグメント間の相互作用エネルギーが全て出力されているので、*PaicsView* を使わず、自身のスクリプトなどで解析してもよい。) 手順は以下の通り。

1. gui window の tool メニューから energy を選択し、paics energy window を開く。
2. 一番上のテキストボックスに、1729 個の原子と 106 個の fragment が存在することが表示される。
3. また、calculation result のリストボックスは、

mon-eng rhf	:	106
mon-eng mp2(ri)	:	106
distance	:	5565
i f i e rhf	:	5565
i f i e mp2(ri)	:	5565

となっている。これは、RHF および RI-MP2 のモノマーエネルギーが 106 個読まれ、fragment 間の距離、RHF および RI-MP2 の IFIE (フラグメント間の相互作用エネルギー) が 5565 個読まれていることを意味する。以下の計算から分かるように、5565 は、106 個の fragment の全組み合わせである。

$$106 * ( 106 - 1 ) / 2 = 5565$$

4. energy sum のチェックボックスのうち、rhf と mp2(ri) をオンにする。これは、以降の解析で、RHF エネルギーと RI-MP2 エネルギーの和を使うことを指定している。( どちらか一方のエネルギーのみを解析したい場合は、一方だけをオンにする。) また、今回は、通常のカノニカル MP2 を実行していないので、mp2 のチェックボックスは選択不可になっている。
5. 次に、fragment を 2 のグループに分ける。以下では、ここで分けたグループ間の相互作用が解析されることになる。具体的には、

1-102

と入力し、fragment number 2 のテキストボックスに、

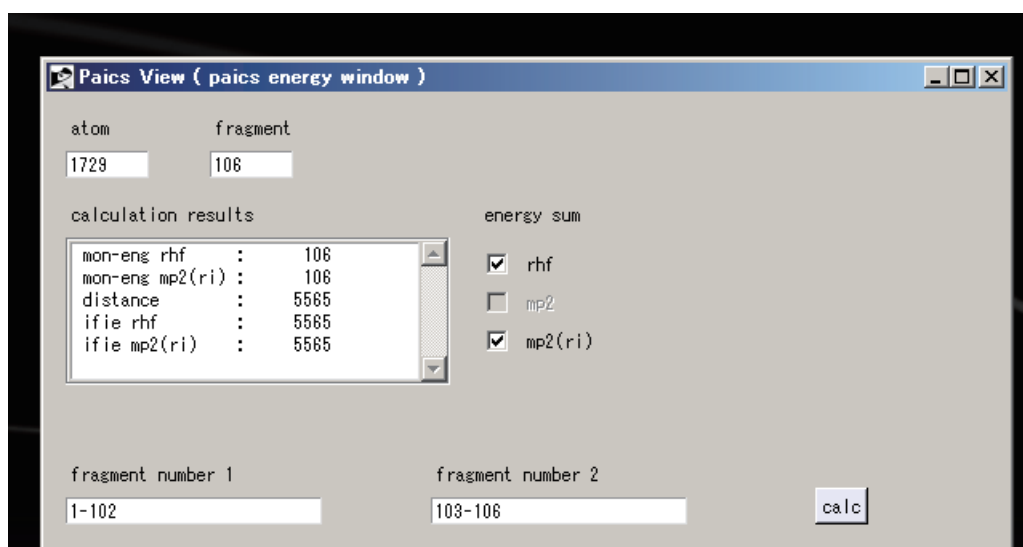
103-106

と入力する。1 番目 ~ 102 番目がプリオンタンパク質の fragment で、これらが 1 つめのグループとなる。一方、103 番目 ~ 106 番目が GN8 の fragment であり、これらが 2 つめのグループとなる。

6. この段階で、paics energy window の上の部分は、図 10 のようになっている。
7. ここで、calc ボタンを押すと、paics energy window の下の部分は図 11 のようになる。それぞれの出力は、以下の通り。
  - internal energy 1 (a.u.)  
1 つめのグループ ( 1 番目 ~ 102 番目の fragment で、プリオンタンパク質に対応する部分 ) の内部エネルギー。単位は原子単位。
  - internal energy 2 (a.u.)  
2 つめのグループ ( 103 番目 ~ 106 番目の fragment で、GN8 に対応する部分 ) の内部エネルギー。単位は原子単位。
  - interaction energy (kcal/mol)  
2 つのグループ間の相互作用エネルギー。単位は kcal/mol。
8. また、グループ間の相互作用エネルギーは、fragment ごとに分割されリストボックスに出力される。例えば、左側のリストボックスの一番上の行は、

1      1.003    8.31    GLY124

図 10: paics energy window の上の部分。



となっている。これは、1 番目の fragment が、

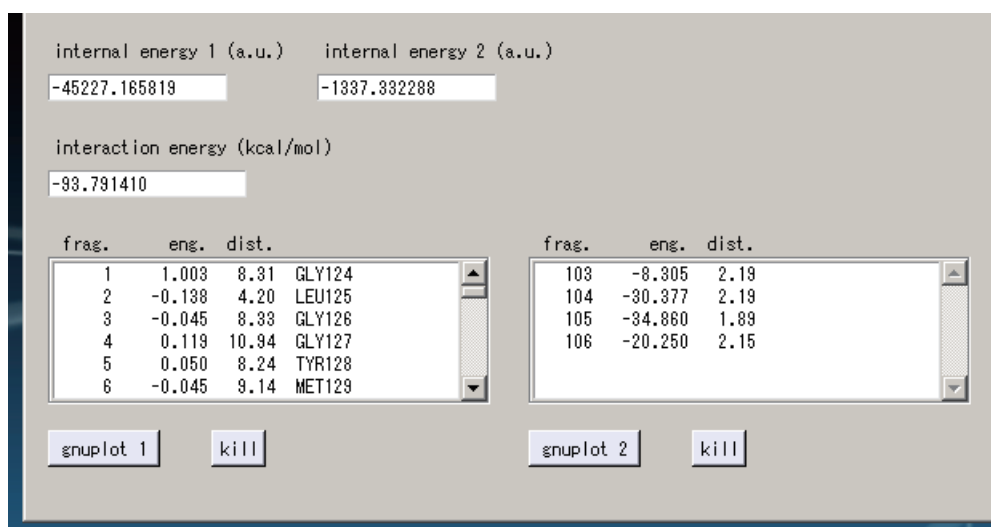
- (1) 相手のグループ (GN8) と 1.003 kcal/mol の相互作用エネルギーをもつ
- (2) 相手のグループ (GN8) と 8.31Å 離れている
- (3) GLY124 の biounit に対応している

ことを示している。同様に、右側のリストボックスは、GN8を構成する4つのfragmentが、相手のグループ (プリオンタンパク質) と及ぼす相互作用エネルギーを示している。これらを解析することで、プリオンタンパク質とGN8の結合において、「どのアミノ酸残基が重要に寄与しているか」もしくは「GN8のどの部位が重要に寄与しているか」を調べることができる。具体的な解析例を以下に示す。

- 左側のリストボックスで、相手のグループとの距離が 3.0 Å 以内のフラグメントを抜き出すと、

7	-0.649	2.15	LEU130
13	-1.830	2.18	ARG136
33	-2.875	2.42	ARG156
34	-2.572	2.52	TYR157
35	-8.490	2.18	PRO158
36	-14.143	1.89	ASN159
37	-11.805	2.49	GLN160
38	-1.511	2.69	VAL161
39	-1.416	2.64	TYR162
60	-1.867	2.19	THR183

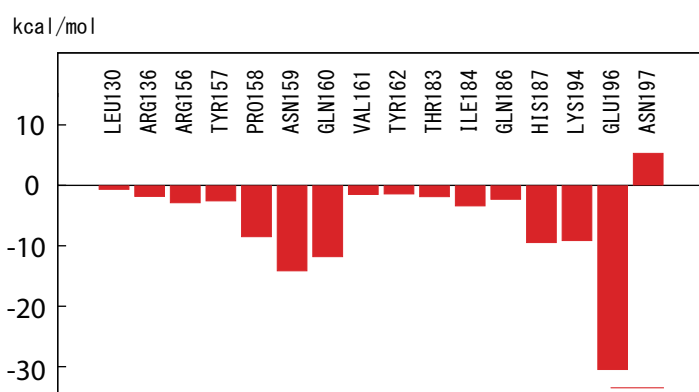
図 11: paics energy window の下の部分 (calc ボタンを押した後)。



61	-3.398	2.80	ILE184
63	-2.305	2.58	GLN186
64	-9.468	2.44	HIS187
71	-9.140	2.38	LYS194
73	-30.473	2.19	GLU196
74	5.290	2.65	ASN197

となる。これは、GN8 から 3.0 Å 以内に存在するアミノ酸残基が、GN8 と及ぼす相互作用エネルギーに対応する。この数値をプロットすると、図 12 となる。

図 12: 相互作用エネルギーのプロット (1)



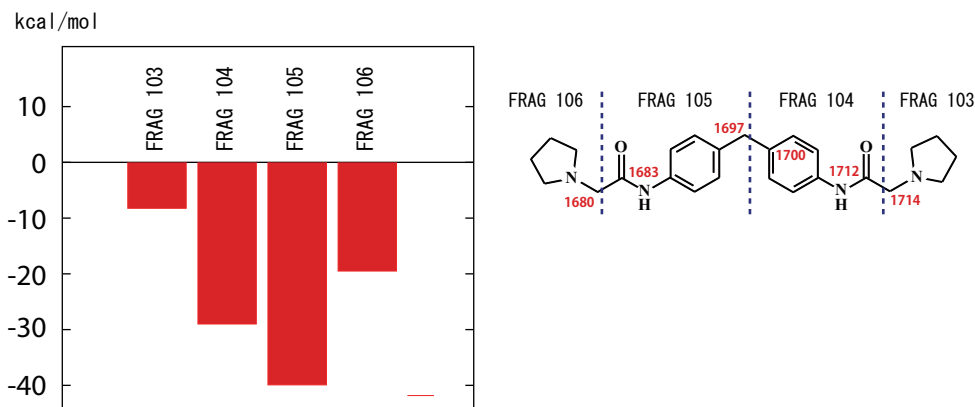
このデータから、プリオンタンパクが GN8 と結合する際、ASN159、GLN160、HIS187、LYS194、GLU196 のアミノ酸残基が、重要な働きをしていると考えられる。

- 右側のリストボックスは、

103	-8.305	2.19
104	-30.377	2.19
105	-34.860	1.89
106	-20.250	2.15

となっている。これは、GN8 の 4 つの部位が、プリオンタンパクと及ぼす相互作用エネルギーに対応する。この数値をプロットすると、図 13 となる。

図 13: 相互作用エネルギーのプロット (2)



このデータから、GN8 のうち、真ん中の部位はプリオンタンパクとの結合に重要に寄与し、両端の部分は、それほど大きく寄与しないと考えられる。

#### [ 注意 1 ]

量子化学計算で評価されるのは、結合自由エネルギーのエンタルピー項のみである。また、今回の例では、溶媒分子も含まれていない。さらに、古典計算のトラジェクトリーから抜き出した、たった 1 つの構造での計算である。生体分子に量子化学計算を応用する場合は、これらの事情を十分考慮し、結果を適切に解釈する必要がある。

9. その下にある、gnuplot ボタンおよび kill ボタンは、それぞれのリストボックスに出力されている相互作用エネルギーを図示するためのものであるが、ここでは説明を省略する。(この機能を使用するためには、gnuplot の設定が必要となる)