作成:2024 年 11 月 18 日
改定:2026 年 03 月 18 日

YASARA チュートリアル ループモデリング

株式会社アフィニティサイエンス

概要:YASARA Structure(Version24.10.5)による、タンパク質構造の欠損ループや欠損末端のモデリング方法について、チュートリアル形式で説明します。

1. はじめに

YASARA では、BuildLoop コマンドを実行することで、タンパク質の欠損したループや末端をモデリングすることができます。ループ(または末端)は、PDB の検索に基づいて適切なモデルが構築されます。本チュートリアルでは、ループと末端が欠損している PDB ファイル(PDB ID:2AC3)を読み込み、欠損箇所の構築、構造全体のリファインなどの操作を行います。

1.1 本チュートリアルの流れ

本チュートリアルの内容は以下になります。

全体の流れ

2. 準備

ループモデリングを行いたい構造を読み込み、簡単な前処理を行います。今回は、ヒトの Mnk2 キナーゼドメインの構造、PDB ID:2AC3 の構造を使用します。

2.1 PDB ダウンロードと構造表示

メニューから File > Load > PDB file from Internet を選択し、PDB ID に「2AC3」と入力、右側の「Include residues only present in SEQRES」にチェックを入れて(任意)、PDF ファイルをダウンロードすると、画面上にタンパク 3D 構造が表示されます。

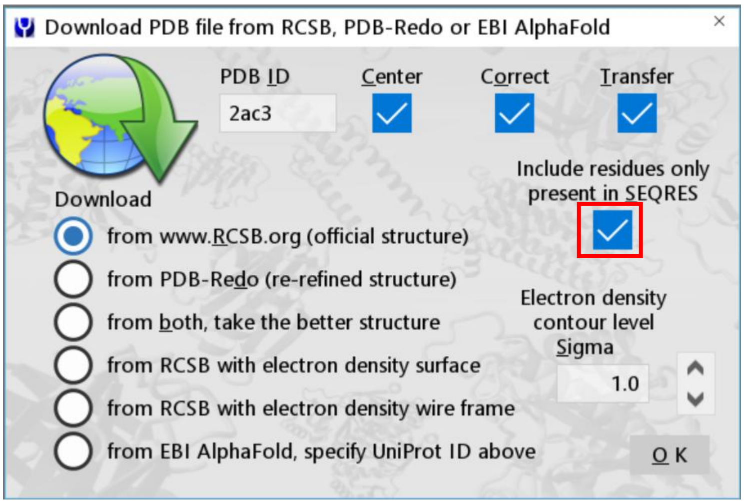

Figure 2-1 PDB ファイルのダウンロード

「Include residues only present in SEQRES」を有効にすると、SEQRES レコード上に存在するものの構造中には存在しない残基を含めて読み込まれます。無効のままでも問題ありませんが、この後の「2.3 欠損残基の確認」の際に便利なので、ここでは有効にしておきます(LoadPDB コマンドの「SeqRes = Yes」オプションに該当します。)。
すると、下の図のように、欠損している残基の C α が水色の球で表現されます。

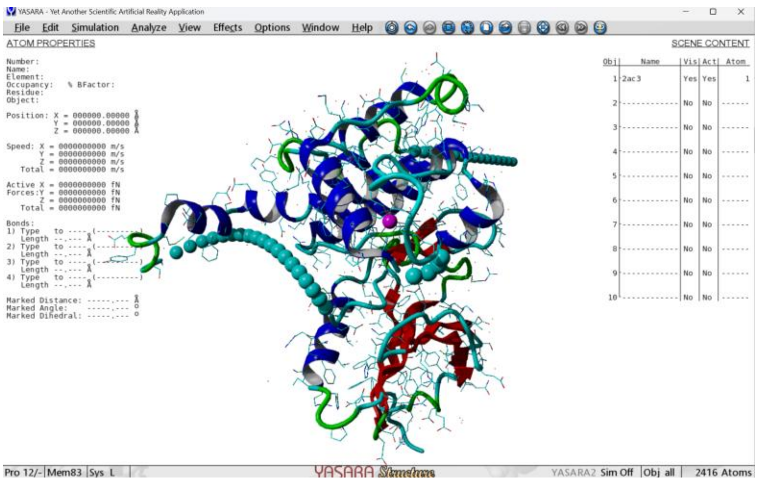

Figure 2-2 SeqRes=Yes により欠損残基の Cα が補完された構造

2.2 前処理(不要分子の削除)

この操作はループモデリング後でも問題ありませんが、ここで不要な分子を削除しておきます。右側の HUD「SCENE CONTENT」からオブジェクトの内容を確認します。

今回読み込んだオブジェクトは水分子のみが不要な分子として含まれているので、Edit > Delete > Waters を選択し、水分子を一括削除します。

【Tips】不要分子が多数ある場合の処理方法
　Edit > Select などから必要な分子を選択状態にしておき、Edit > Delete の選択ダイアログで Belongs to or has リストの「Selected」を選択し、リスト下の「Negate attribute」にチェックを入れて有効にして OK をクリックすると、選択した分子以外を一括削除できます。

2.3　欠損残基の確認

　次に、欠損している残基番号を確認します。決まった方法はありませんが、いくつかの方法を以下にご紹介するので、これらを参考に欠損している残基の番号を確認してください。

❑　「List」コマンドで確認する方法

　コマンドを使って操作するので、Space キーを押して、コンソール画面を開いてください。コンソール画面を開いたら、次のコマンドを入力し、Enter キーを押して実行します。（※PDB ファイルの読み込み時に、「Include residues only present in SEQRES」を有効にしていることが前提となります。）

　ListAtom CA Occupancy=0

　すると、以下のように欠損している残基（Occupancy＝0 の Cα原子）のリストが出力されます。

```
>ListAtom CA Occupancy=0
Atom 1271  CA  ASP  226  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 72.74, Prop 0.000, X 4.287, Y -3.437, Z 0.030
Atom 1279  CA  PHE  227  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 81.85, Prop 0.000, X 6.278, Y -5.528, Z 2.513
Atom 1290  CA  ASP  228  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 78.45, Prop 0.000, X 8.863, Y -2.781, Z 2.761
Atom 1315  CA  GLY  232  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 14.278, Y 4.043, Z 3.011
Atom 1316  CA  ILE  233  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 14.239, Y 6.305, Z 3.503
Atom 1317  CA  LYS  234  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 13.981, Y 8.099, Z 3.322
Atom 1318  CA  LEU  235  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 13.548, Y 9.473, Z 2.543
Atom 1319  CA  ASN  236  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 12.985, Y 10.476, Z 1.244
Atom 1320  CA  GLY  237  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 12.337, Y 11.157, Z -0.499
Atom 1321  CA  ASP  238  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 11.649, Y 11.567, Z -2.610
Atom 1322  CA  CYS  239  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 10.966, Y 11.752, Z -5.012
Atom 1323  CA  SER  240  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 10.333, Y 11.764, Z -7.629
Atom 1324  CA  PRO  241  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 9.795, Y 11.650, Z -10.385
Atom 1325  CA  ILE  242  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 9.396, Y 11.460, Z -13.202
Atom 1326  CA  SER  243  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 9.181, Y 11.243, Z -16.005
Atom 1327  CA  THR  244  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 9.196, Y 11.048, Z -18.717
Atom 1328  CA  PRO  245  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 9.485, Y 10.923, Z -21.262
Atom 1329  CA  GLU  246  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 10.093, Y 10.919, Z -23.562
Atom 1330  CA  LEU  247  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 11.065, Y 11.084, Z -25.543
Atom 1331  CA  LEU  248  A: C , Obj 1 (2ac3), AltLoc  , Occ 0.00, BFac 0.00, Prop 0.000, X 12.446, Y 11.467, Z -27.126
```

Figure 2-3　ListAtom コマンド出力画面の一部

　確認してみると、**232-250**、**306-309**、**370-385** の残基番号が欠損していることがわかります。（※残基番号 226-228 は実際には欠損していないので、除いています。）

（解説）

　PDB ファイルの読み込み時に、「Include residues only present in SEQRES」を有効にしておくと、欠損している残基の情報も含めて構造が表示されます。このとき、欠損している部分の残基は、Cα原子のみが水色の球として表示され、Occupancy の値は0に設定されます。そのため、「List」コマンド（指定した原子や分子などをリスト表示するコマンド）で、Occupancy＝0 の Cα原子を出力することで、欠損している残基をリスト表示することができます。

　なお、欠損していない残基でも、信頼性が低い場合に Occupancy が0に設定されている場合があり、リストに出力されることがあります（今回の例では、残基番号 226-228）。この場合は、YASARA の 3D ビュー画面で構造を確認する、または、Bfactor 値が出力されているか、PDB ファイル（テキストエディタで開くことができます）の「ZERO OCCUPANCY RESIDUES」の項目に記載されているか等を確認して見分けてください。

❑ プロテインデータバンクで確認する方法

RCSB プロテインデータバンク(https://www.rcsb.org/)にアクセスし、右上の検索窓に「2AC3」と入力して検索します。

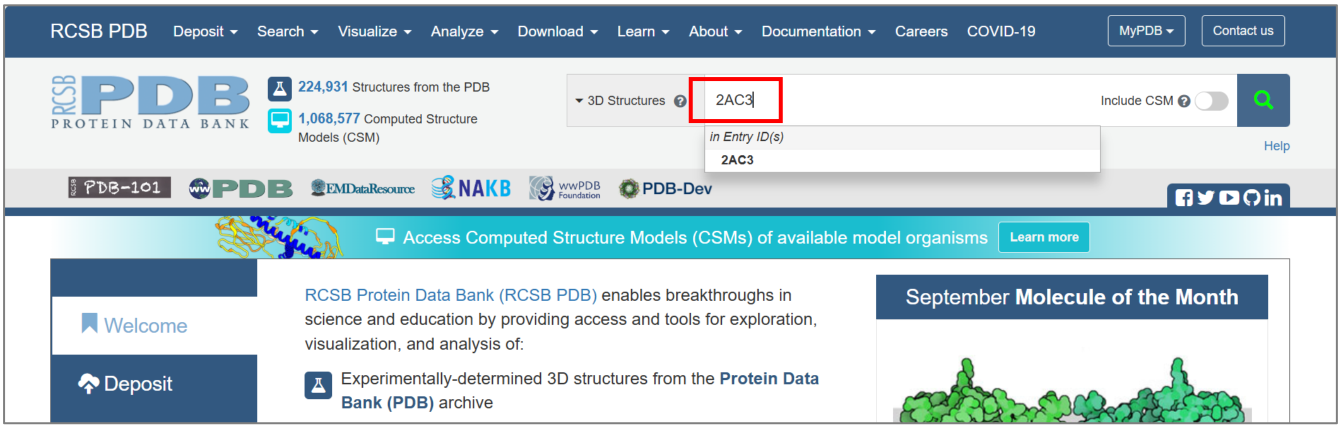

Figure 2-4 プロテインデータバンクの HP

すると、PDB ID:2AC3 のページが開くので、「Sequence」タブを選択すると、シーケンスの詳細情報を確認できます。上から4つ目の「UNMODELED」の項目の、グレーの部分が欠損残基になります。グレーの部分にマウスオーバーすると、右上に該当する残基の番号が表示されます。右端の「auth:~」が、YASARA 内での残基番号と一致するので、こちらを控えておきます。

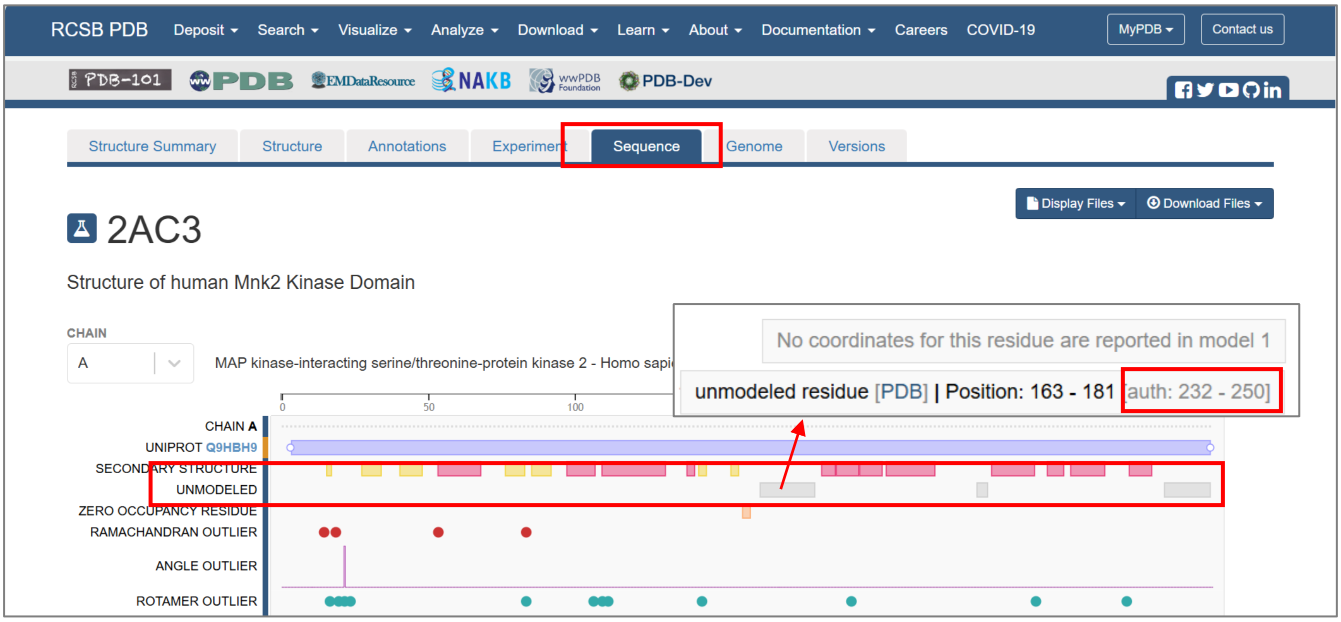

Figure 2-5 Sequence タブ

❑ PDB ファイルで確認する方法

前述の手順 3.1.2 で記載したように、RCSB プロテインデータバンクで PDB ID:2AC3 のページにアクセスします。右側にある「Download Files」をクリック、「PDB Format」を選択し、PDB ファイルをダウンロードします。

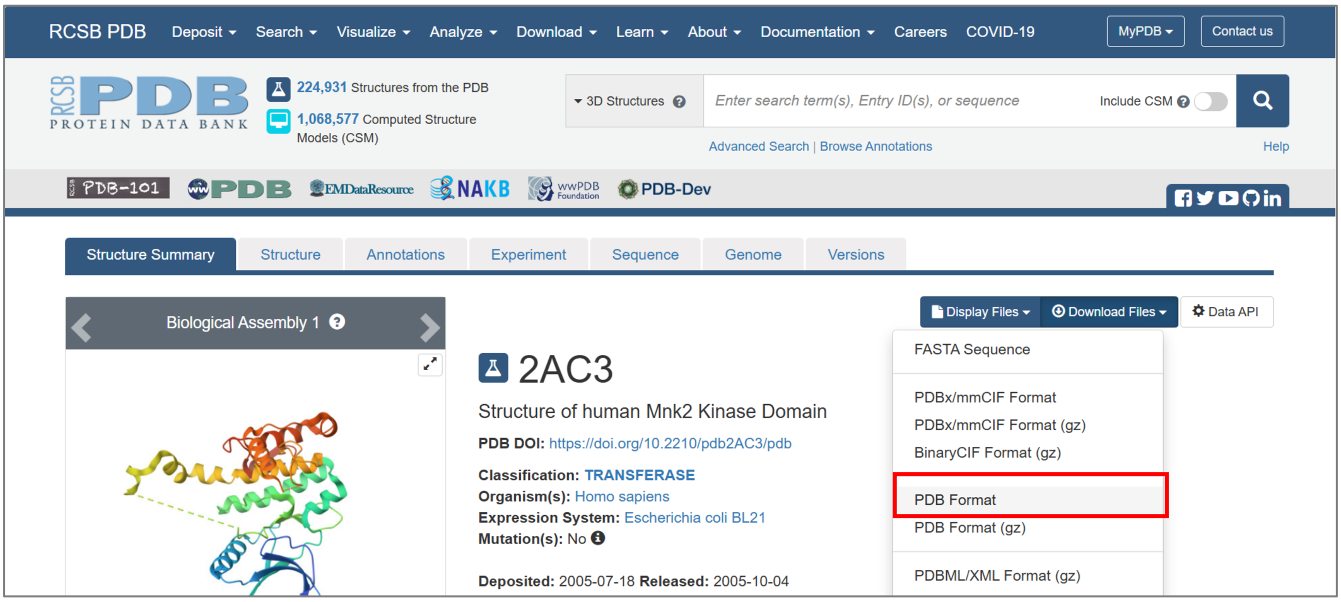

Figure 2-6 PDB ファイルのダウンロード

次に、ダウンロードした PDB ファイルをメモ帳などのテキストエディタで開きます。Ctrl + F キーなどで「REMARK 465 MISSING RESIDUES」セクションを検索すると、欠損残基の情報が確認できるので、残基番号を控えておきます。

```
REMARK 465
REMARK 465 MISSING RESIDUES
REMARK 465 THE FOLLOWING RESIDUES WERE NOT LOCATED IN THE
REMARK 465 EXPERIMENT. (M=MODEL NUMBER; RES=RESIDUE NAME; C=CHAIN
REMARK 465 IDENTIFIER; SSSEQ=SEQUENCE NUMBER; I=INSERTION CODE.)
REMARK 465
REMARK 465   M RES C SSSEQI
REMARK 465     GLY A   232
REMARK 465     ILE A   233
REMARK 465     LYS A   234
REMARK 465     LEU A   235
REMARK 465     ASN A   236
REMARK 465     GLY A   237
REMARK 465     ASP A   238
REMARK 465     CYS A   239
REMARK 465     SER A   240
REMARK 465     PRO A   241
REMARK 465     ILE A   242
REMARK 465     SER A   243
REMARK 465     THR A   244
REMARK 465     PRO A   245
REMARK 465     GLU A   246
REMARK 465     LEU A   247
REMARK 465     LEU A   248
REMARK 465     THR A   249
REMARK 465     PRO A   250
REMARK 465     ASP A   306
REMARK 465     ARG A   307
REMARK 465     GLY A   308
REMARK 465     GLU A   309
REMARK 465     GLY A   370
REMARK 465     CYS A   371
REMARK 465     ALA A   372
REMARK 465     PRO A   373
REMARK 465     GLU A   374
REMARK 465     ASN A   375
REMARK 465     THR A   376
```

Figure 2-7 PDB ファイルの「REMARK 465 MISSING RESIDUES」セクション

❑ Sequence Selector から確認する方法

YASARA で PDB ファイルをダウンロードする際に「Include residues only present in SEQRES」を有効にしていると、欠損残基の配列情報も Sequence Selector に表示されます。欠損残基（正確には Occupancy＝0の残基）は、グレーで表示され、二次構造情報も欠けているので、ここから確認することもできます。（※残基番号 226-228 は Occupancy が0のためグレーで表示されていますが、欠損している訳ではないので注意してください。）

「Include residues only present in SEQRES」を無効にして読み込んだ場合、残基番号が飛んでいる部分を探すことで欠損残基確認することもできますが、末端が欠損している場合は確認できないので注意が必要です。

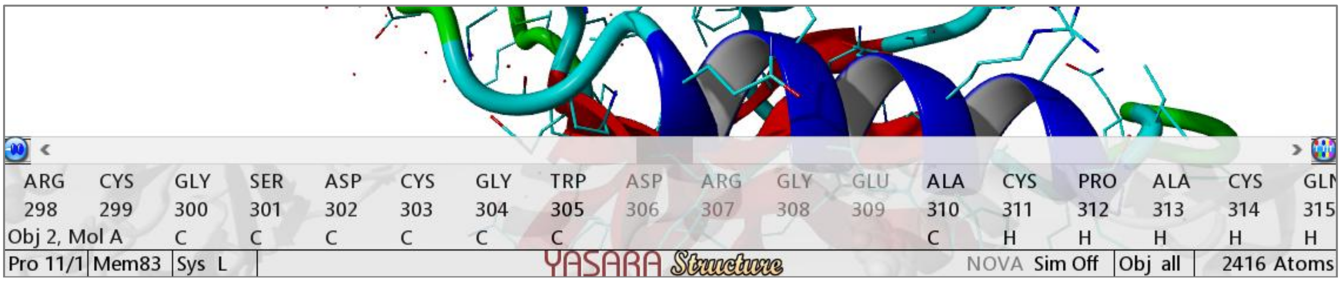

Figure 2-8 Sequence Selector の表示例（残基番号 306-309 が欠損している）

2.4 シーケンス（一文字コード）の取得

アミノ酸配列は YASARA の Sequence Selector や PDB ファイルからも確認できますが、次のループモデリング実行時には欠損配列をアミノ酸の一文字コードで指定するため、事前に一文字コードの配列情報を取得しておくとよいです。以下の例を参考に、fasta ファイルをダウンロードしたり、YASARA でアミノ酸の一覧を出力するなどして、配列情報を取得します。

❑ fasta 形式ファイルを取得する

手順 2.3 の「欠損残基の確認」を参考に、RCSB プロテインデータバンクのページから「Download Files」を選択、一番上の「Fasta Sequence」をクリックして fasta ファイルをダウンロードし、適当な場所に保存しておいてください。fasta ファイルはメモ帳などのテキストエディタで開くことができます。

❑ YASARA でアミノ酸配列を出力する

SequenceRes コマンドを利用することで、アミノ酸配列を一文字コードで出力することができます。メニューから Analyze > Sequence of > Residue を選択し、右の Belongs to or has リストから AminoAcid を選択するなどして実行すると、コンソール画面に配列が出力されます。（対象の選択については、モデルに応じて適宜対応してください。）

```
>SequenceRes AminoAcid
Object 1 (2ac3), selected residues in molecule A:
GSTDSFSGRFEDVYQLQEDVLGEGAHARVQTCINLITSQEYAVKIIEKQPGHIRSRVFREVEMLYQCQGHRNVLELIEFFEEEDRFYLVFEKMRGGSILSHIHKRRHFNELEASVV
VQDVASALDFLHNKGIAHRDLKPENILCEHPNQVSPVKICDFDLGSGIKLNGDCSPISTPELLTPCGSAEYMAPEVVEAFSEEASIYDKRCDLWSLGVILYILLSGYPPFVGRCGS
DCGWDRGEACPACQNMLFESIQEGKYEFPDKDWAHISCAAKDLISKLLVRDAKQRLSAAQVLQHPWVQGCAPENTLPTPMVLQR
>
```

Figure 2-9 SequenceRes コマンド出力例

2.5 欠損領域とアンカーの情報の整理

この後の欠損残基のモデリングには BuildLooop コマンドを実行しますが、その際に必要な情報をここで整理しておきます。コマンドの実行時に必要な情報は、①開始と終了のアンカー（通常 2~3 残基分の残基番号と残基の一文字コード）と、②欠損配列（一文字コード）、になります。

先ほど取得したシーケンス情報を利用するなどして、欠損配列の文字列（一文字コード）や前後のアンカーを確認してください。

（RCSB のサイトで PDB ID:2AC3 のページを開き、「Structure」タブを開くと、配列情報が表示されます。欠損部分が灰色で表示されているので、ここから欠損配列部分を確認することもできます。ただし、欠損配列だけではなく、構造情報があるものの占有率が 0 に設定されている残基（例：Res226-228）についても灰色で表示されるので注意してください。）

以下は、シーケンスの Fasta ファイルを着色したものです。黄色がアンカー（アンカーには通常 2~3 残基指定します）、灰色が欠損配列となっています。

```
>2AC3_1|Chain A|MAP kinase-interacting serine/threonine kinase 2|Homo sapiens (9606)
GSTDSFSGRFEDVYQLQEDVLGEGAHARVQTCINLITSQEYAVKIIEKQPGHIRSRVFREVEMLYQCQGHRNVLELIEFFE
EEDRFYLVFEKMRGGSILSHIHKRRHFNELEASVVVQDVASALDFLHNKGIAHRDLKPENILCEHPNQVSPVKICDFDLG
SGIKLNGDCSPISTPELLTPCGSAEYMAPEVVEAFSEEASIYDKRCDLWSLGVILYILLSGYPPFVGRCGSDCGWDRGEAC
PACQNMLFESIQEGKYEFPDKDWAHISCAAKDLISKLLVRDAKQRLSAAQVLQHPWVQGCAPENTLPTPMVLQR
```

3 か所の欠損部分を整理すると、以下のようになります。()内は残基番号です。

開始のアンカー	欠損配列	終了のアンカー
…LGS (…Res 229,230,231)	GIKLNGDCSPISTPELLTP (Res 232-250)	CGS… (Res 251,252,253…)
…CGW (…Res 303,304,305)	DRGE (Res 306-309)	ACP… (Res 310,311,312…)
…WVQ (…Res 378,368,369)	GCAPENTLPTPMVLQR (Res 370-385)	なし

Table 2-1 欠損領域と周囲のアンカー情報

3. 欠損部位のモデリング（BuildLoop）

アミノ酸残基が欠損している部分の構造をモデリングしていきます。YASARA では、BuildLoop コマンドを使ってミッシングループをモデリングすることができます。また、同じコマンドで、N 末端や C 末端の欠損箇所も構築することができます。
それでは、手順 2.5 で確認した 3 か所の欠損部位を順にモデリングしていきます。

3.1 ミッシングループのモデリング

はじめに、BuilLoop コマンドの使用方法を紹介しながら、一つ目のミッシングループ、**Res 232-250** 部分を構築してみます。BuilLoop コマンドは、PDB データベースから類似の構造を検索し、構造ファイルの欠損したループや末端をモデリングできるコマンドです。（詳細は、ユーザーマニュアルの BuilLoop コマンドページに記載されているので、ご参照ください。）
BuilLoop コマンドは、コンソールからコマンドで実行するほかに、メニューからも実行できるので、それぞれの実行方法を説明します。どちらで実行しても基本的には同じ処理が行われるので、お好みの方法でモデリングしてください。

実行前に、まずはミッシングループ「Res 232-250」の配列やアンカー部分の情報を整理しておきます。

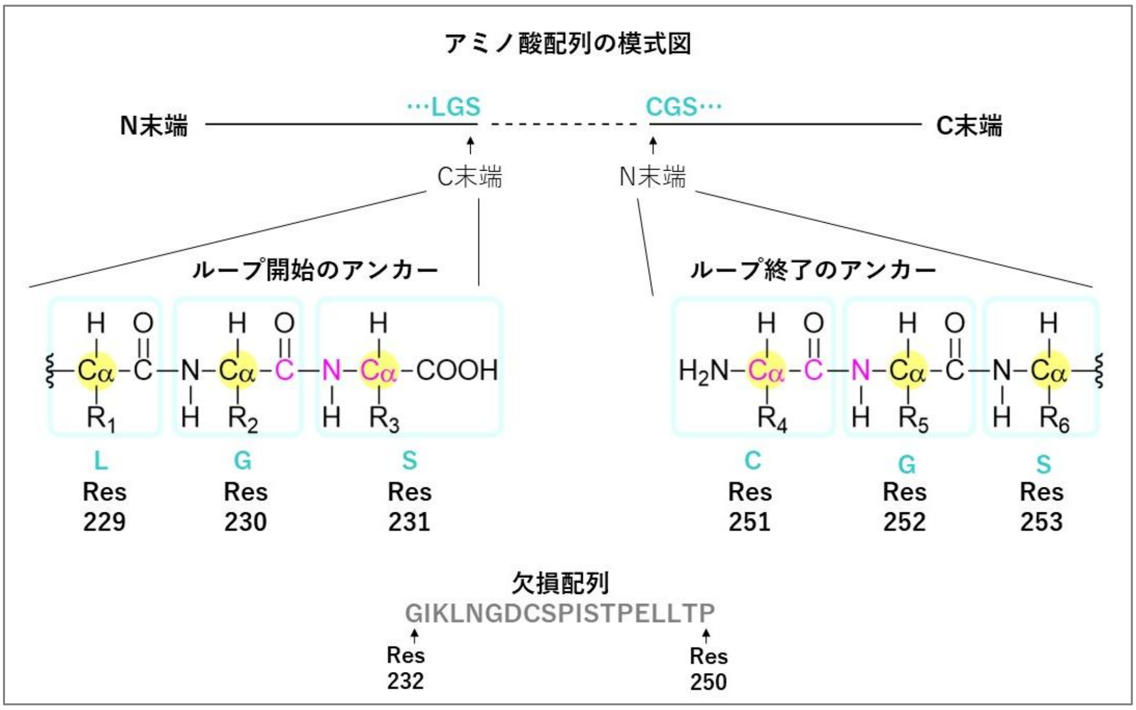

Figure 3-1 ミッシングループ「Res 232-250」の配列とアンカー

まず、BuidLoop コマンドの実行時に必要な情報をまとめておきます。

- ❑ ループ開始・終了のアンカー原子

  欠損配列と接続する部分の、バックボーンとなる原子をそれぞれ 3 つ以上指定します。例えば、Figure 3-1 のピンク色で示したバックボーン原子や、黄色で示した CA炭素などを指定します。

【Note】PDB の即時検索の適用について
アンカー原子に Figure 3-1 のピンク色で示した末端 3 つ分のバックボーン原子を選択（コマンドの場合は開始のアンカーを「C-N-CA」、終了のアンカーを「CA-C-N」の形式で指定）した場合、欠損部分が 17 残基以内であれば PDB のクイックアクセスインデックスを使用した即時検索が適用されます。ただし、他のバックボーン原子を指定した場合には、この即時検索は適用されません。

以下に、コマンド実行時の指定例をいくつか示します。コマンドの入力例で記載しますが、メニューから実行する場合は、それぞれ対応する原子を選択すればよいです。

方法1） 即時検索が可能な指定例（ピンク色で示した原子を指定）

開始アンカー原子：**C Res 230 or N CA Res 231**　終了アンカー原子：**CA C Res 251 or N Res 252**

方法2） Cα炭素を 3 つ指定した例（黄色でハイライトした原子を指定）

開始アンカー原子：**CA Res 229-231**　終了アンカー原子：**CA Res 251-253**

方法3） 2 残基分のバックボーン原子をまとめて指定した例

開始アンカー原子：**Backbone Res 230-231**　終了アンカー原子：**Backbone Res 251-252**

- ❑ 欠損配列（アンカーを含む）

　欠損配列には、アンカーに指定した原子が属する残基を含めて一文字コードで指定します。残基の並びは、タンパク質全体の N 末端→C 末端の配列の方向に合わせて入力します。（通常は残基番号が若い方から順に指定します。）
今回の例では、欠損している Res 232-250 の配列は、前に確認した通り「GIKLNGDCSPISTPELLTP」です。コマンド実行時には、これにアンカーに指定した残基を加えて指定します。

例えば、上記の**方法1**や**方法3**のようにアンカー原子を指定した場合は、アンカー原子が 2 残基分に属しているので、それらを含めて以下のように指定します。（黄色はアンカー残基）

GSGIKLNGDCSPISTPELLTPCG

アンカー原子を**方法2**のように指定した場合は、アンカー原子が 3 残基分に属しているので、以下のように欠損配列を指定します。（黄色はアンカー残基）

LGSGIKLNGDCSPISTPELLTPCGS

3.1.1 メニューから実行する場合

　それでは実際に、ループモデリングを実行します。まずは、メニューから一つ目のミッシングループ、**Res 232-250** 部分を例に、ミッシングループを構築する方法を説明します。

- ❑ メニューから Edit > Build > Loop を選択

　ループを構築したい場合は「Loop」、N 末端を構築したい場合は「N-terminal loop」、C 末端を構築したい場合は「C-terminal loop」をそれぞれ選択します。今回は、ループを構築したいので、「Loop」を選択します。

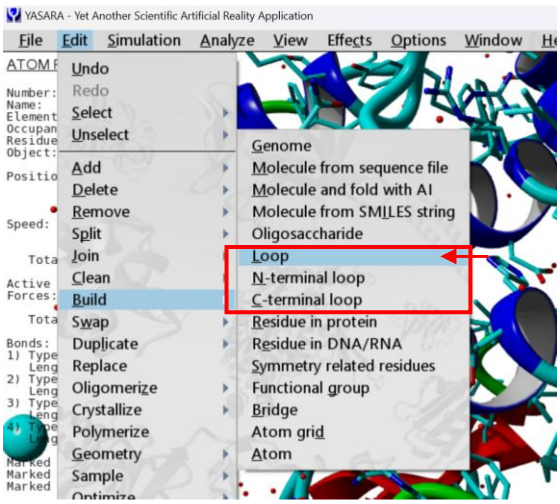

Figure 3-2 ループモデリングメニュー

❑ 開始アンカー原子の選択

すると、アンカー原子の選択画面が表示されるので、ループ開始のアンカー原子(タンパク質の C 末端側)を選択します。指定方法は、前述の「3.1 ミッシングループのモデリング」を参照してください。ここでは例として、残基番号 229-231 の CA 原子を指定し、アンカー原子を Ctrl キーを押しながら複数選択します。OK をクリックして次の画面に進みます。

(表示されるダイアログには、N-terminal anchor と記載されていますが、欠損配列の N 末端側アンカーという意味です。)

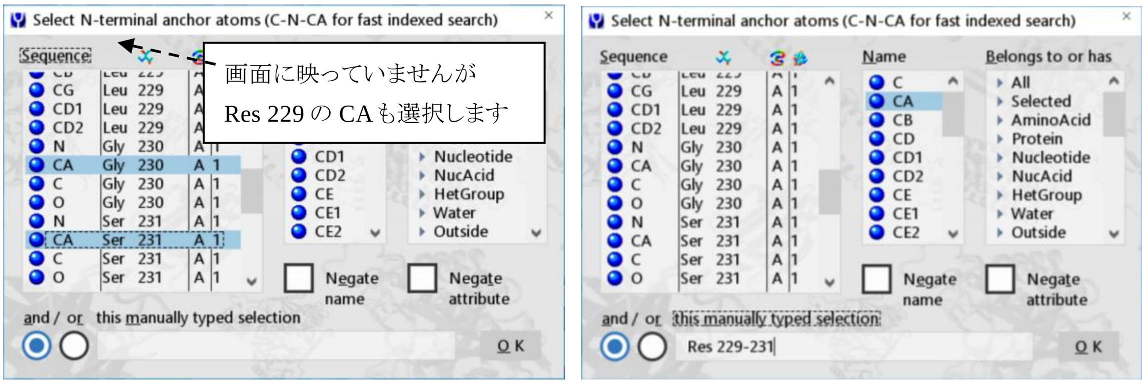

Figure 3-3 (左) アンカー指定方法 2 で α 炭素を 3 つ指定 (右) Name CA と残基番号を用いた指定例

❑ 終了アンカー原子の選択

開始アンカー原子と同じように、終了アンカーとなる残基番号 251-253 の CA 原子を指定し、OK をクリックします。

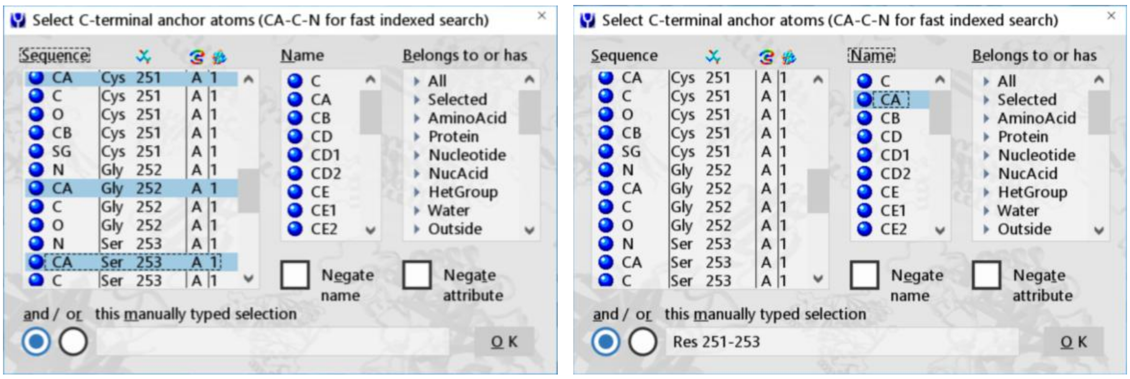

Figure 3-4 (左) アンカー指定方法 2 で α 炭素を 3 つ指定 (右) 残基番号 251-253 の CA 原子を指定した例

❑ アンカーを含む欠損配列の入力とオプション設定

「Sequence in one letter code…」で始まる入力欄に、欠損配列を一文字コードで入力します。アンカーの残基を含めることに注意し、前述の「3.1 ミッシングループのモデリング」を参考に指定してください。

今回はアンカー原子が 3 残基分に属しているので、以下のように入力します。(黄色はアンカー残基)

LGSGIKLNGDCSPISTPELLTPCGS

また、この画面からはオプションを指定できますが、ここではデフォルト設定のまま、OK をクリックし、モデリングを実行します。オプションの詳細については、補足「6.2 BuildLoop コマンドのオプションについて」をご覧ください。

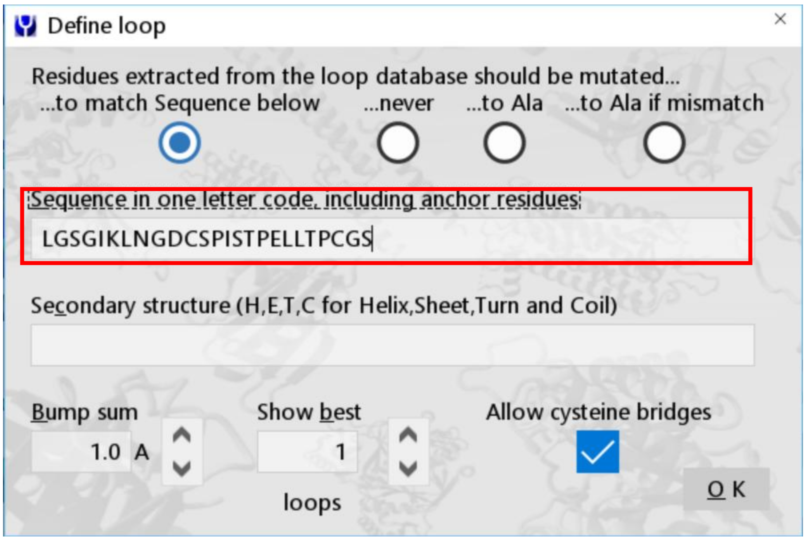

Figure 3-5　配列の入力とオプション設定例（前後 3 残基をアンカーに指定した例）

3.1.2 コマンドで実行する場合

次に、2 つ目のミッシングループ「**Res 306-309**」を例に、コマンドでの実行方法を説明します。前述したように、BuildLoop コマンドは特定の形式でアンカー原子を指定すると PDB の即時検索が適用されるので、今回はこの形式で実行してみます。（メニューから実行する際も、同様のアンカー原子を指定すれば即時検索を利用したモデリングが可能です。）
欠損配列やアンカーについては手順 2.5 でまとめていますので、そちらもご参照ください。

- **BuildLoop コマンドのフォーマットについて**

コマンドは、以下のフォーマットで実行します。

BuildLoop [①ループ開始のアンカー],[②欠損配列(アンカーを含む)],[③ループ終了のアンカー],[④オプション]

※N 末端を構築する場合は、①ループ開始のアンカーに「None」と入力します。
※C 末端を構築する場合は、③ループ終了のアンカーに「None」と入力します。

①③のアンカー原子の指定方法、②の欠損配列の指定方法は、前述の「3.1 ミッシングループのモデリング」をご覧ください。
④のオプションについては、補足「6.2BuildLoop コマンドのオプションについて」にまとめているのでそちらをご参照ください。

以上より、**Res 306-309** のミッシングループを構築するコマンド例は以下のようになります。（黄色はアンカー残基）
方法1 のコマンドは、PDB の即時検索が適用される形式になっているので、今回はこのコマンドを実行してみます。

方法 1) BuildLoop C Res 304 or N CA Res 305,GWDRGEAC,CA C Res 310 or N Res 311
方法 2) BuildLoop CA Res 303-305,CGWDRGEACP,CA Res 310-312
方法 3) BuildLoop Backbone Res 304-305,GWDRGEAC,Backbone Res 310-311

- **コマンドの実行**

準備ができたら以下の手順でコマンドを実行し、ミッシングループを構築します。ここでは例として、PDB の即時検索が適用される方法（方法1）を実行してみます。

- ❑ Space キーを押して、コンソール画面を開きます。
- ❑ 下記のコマンド例をコピーするなどして、コンソール画面にコマンドを入力します。

　　BuildLoop C Res 304 or N CA Res 305,GWDRGEAC,CA C Res 310 or N Res 311

- ❑ Enter キーを押してコマンドを実行します。

```
>BuildLoop C Res 304 or N CA Res 305,GWDRGEAC,CA C Res 310 or N Res 311
Loop  1, object  1: From PDB file 2pqg (resolution 2.38 A).
>
```

Figure 3-6 BuildLoop コマンド実行例

実行すると、コンソール画面に採用された PDB 構造の ID が出力されます。この形式でアンカー原子を指定すると、短い時間でモデリングが可能です。

3.2 欠損した末端のモデリング

ミッシングループが構築できたら、続いて欠損している C 末端（**Res 370-385**）のモデリングを行います。N 末端や C 末端の欠損についても、BuildLoop コマンドを使って同様の操作でモデリングができます。こちらもコンソールからコマンドで実行する方法と、メニューから実行する方法を順に説明します。実行時の処理内容は基本的に同じなので、お好きな方で実行してください。欠損配列やアンカーについては、手順 2.5 でまとめていますので、そちらを確認しながら進めてください。

3.2.1 メニューから実行する場合

基本的にはループモデリングと同じ操作になります。

- ❑ メニューから Edit > Build > C-terminal loop を選択
（N 末端をモデリングしたい場合は、「N-terminal loop」を選択します。）

- ❑ 開始アンカー原子の選択
ループモデリング時と同様の指定方法で開始アンカー原子を選択します。（「N-terminal loop」を選択した場合は、終了アンカー原子を選択します。）
以下に、左図が残基番号 368-369 のバックボーン原子を指定した例（PDB の即時検索が適用される指定方法）、右図が残基番号 367-369 の CA 原子を指定した例をそれぞれ示します。

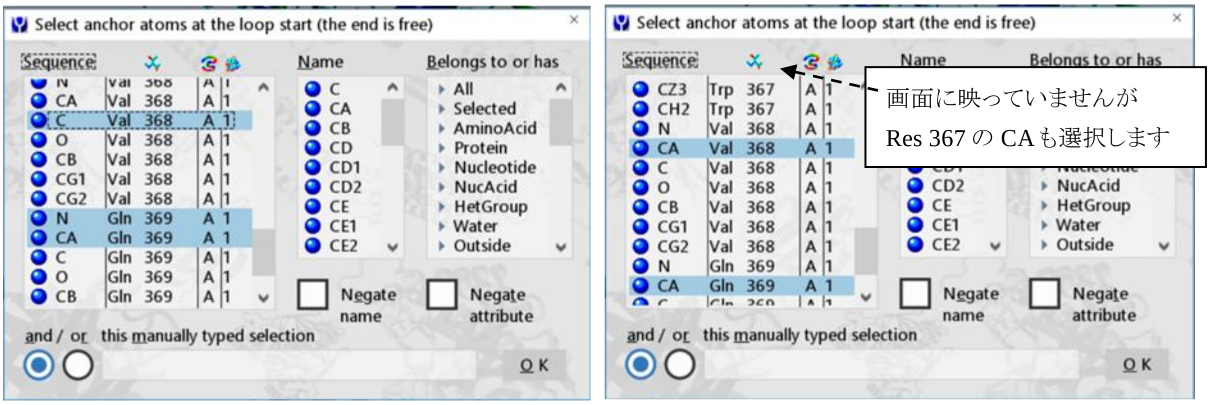

Figure 3-7 (左) 残基番号 368-369 のバックボーン原子を指定した例 (右) 残基番号 367-369 の CA 原子を指定した例

- ❑ アンカーを含む欠損配列の入力とオプション設定
ループモデリングの時と同じように、アンカーを含めて欠損残基の配列を入力し、今回もオプションはデフォルト設定のまま OK をクリックしてモデリングを実行します。

例1） VQGCAPENTLPTPMVLQR （2 残基分の開始アンカーを指定した場合）
例2）WVQGCAPENTLPTPMVLQR （3 残基分の開始アンカーを指定した場合）

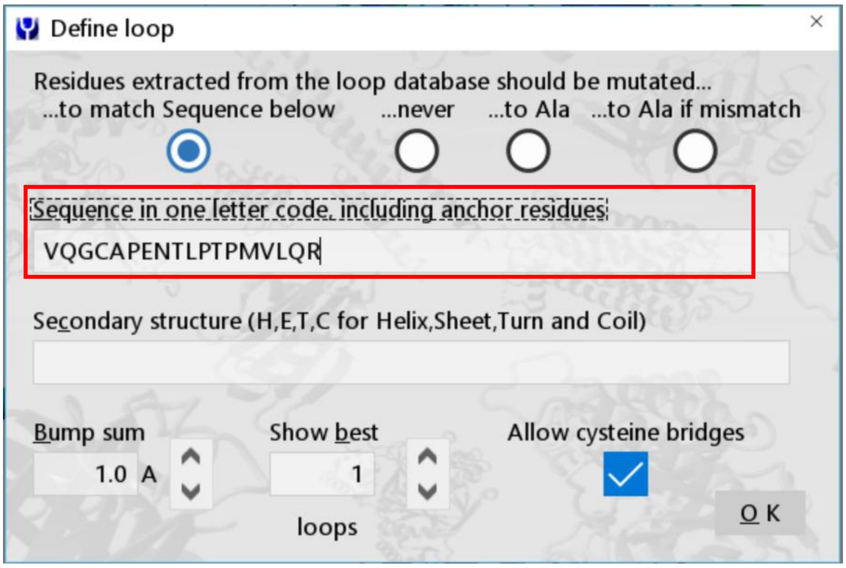

Figure 3-8　配列の入力とオプション設定例（2 残基分を開始アンカーに指定した例）

3.2.2 コマンドで実行する場合

基本的にはループモデリングと同じなので、手 3.1.2 の「コマンドで実行する場合」と同様にコマンドを実行します。ただし、今回は C 末端側が欠損しており、終了のアンカー原子が存在しないため、その部分には「None」と指定します。
（N 末端側が欠損している場合は、開始のアンカー原子の指定部分を「None」とします。）
以下がコマンド例になります（黄色部分はアンカー残基）。なお、例1のコマンド例は PDB の即時検索が適用される形式です。

例1） BuildLoop C Res 368 or N CA Res 369,VQGCAPENTLPTPMVLQR,None
例2） BuildLoop CA Res 367-369,WVQGCAPENTLPTPMVLQR,None
例3） BuildLoop Backbone Res 368-369,VQGCAPENTLPTPMVLQR,None

4. モデリング部分の最適化（OptimizeLoop）

続いて、構築したモデル部分を最適化するコマンド、OptimizeLoop を実行し、モデルの品質向上を試みます。モデルの品質は、Z-スコア（Check コマンド）で比較してみます。

4.1 クリーニング処理

事前にクリーニング処理を行っておきます。以下の Clean コマンドを実行し、不要な分子があれば削除しておきます。

- ❑ Edit > Clean > All

4.2 モデルの品質チェック（最適化前）

これから OptimizeLoop コマンドを使って構築したループや末端のモデルを最適化しますが、実行前後のモデルの品質を比較したいので、Check コマンドを使って最適化前のモデルの Z-スコアを確認しておきます。
事前に力場を YASARA2 に設定してから、Check コマンドを使って現在のモデルの Z-スコアをチェックします。

- ❑ Simulation > Force field を選択し、リストから「YASARA2」を選択して「OK, and~」をクリック

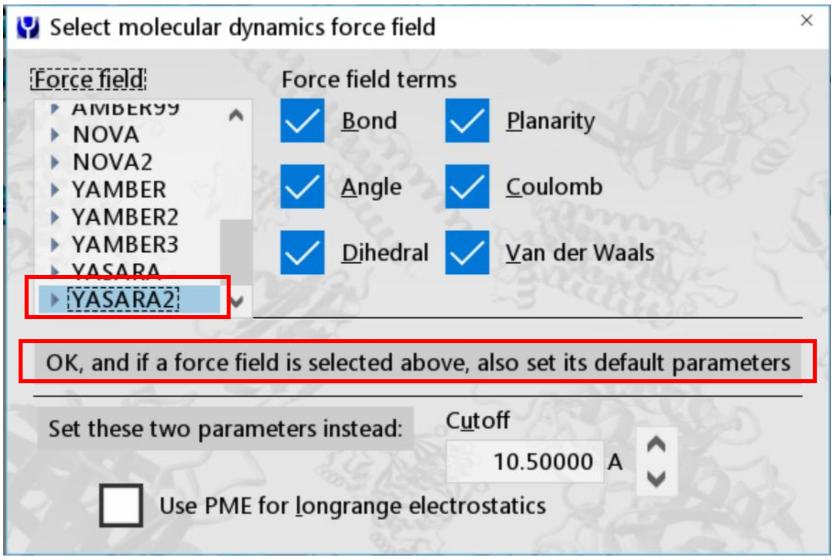

Figure 4-1 力場の設定画面

- ❑ Analyze > Check > All から、「ModelQuality:~」を選択し、OK

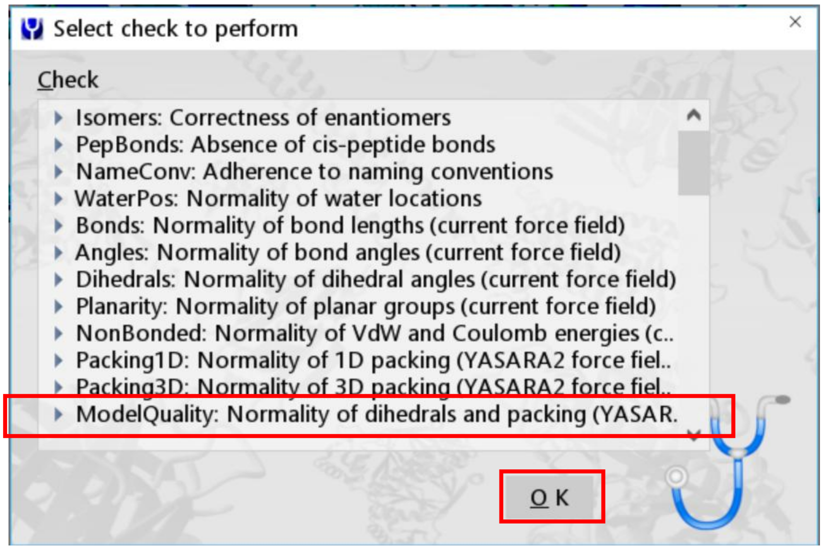

Figure 4-2 Check コマンドの実行画面

すると、コンソール画面に Z-スコアが出力されます。（アンカー原子の指定方法によって多少結果が異なる場合があります。）この例では、最低化前の Z-スコアは -1.895 でした。-

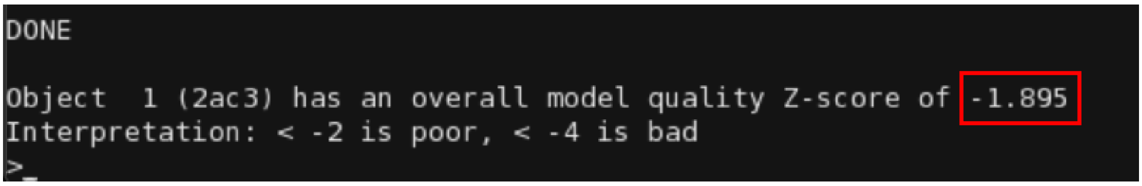

Figure 4-3　Z-スコアの確認①

4.3　モデリングしたループ・末端の最適化

続いて、OptimizeLoop コマンドを実行し、作成した欠損部位の最適化を行います。OptimizeLoop コマンドは、アンカー原子（と、指定した場合は二次構造）を考慮して、指定した数の構造を PDB から抽出し、ループの最適化とエネルギー計算を行い、最もスコアが高かった構造が採用されます。BuildLoop コマンドの実行時と同じようにアンカー原子を指定して実行します。

以下に、コマンドで実行する場合と、メニューから実行する場合の手順を順に紹介します。

※先ほど設定したので操作を省略しますが、OptimizeLoop コマンド実行時は力場を YASARA2 にしておくことが推奨されています。

4.3.1 コマンドで実行する場合

コマンドで実行する場合は、以下のようなフォーマットになります。

```
OptimizeLoop [①ループ開始のアンカー],[②ループ終了のアンカー], [③オプション]
```

①と②のアンカーは、BuildLoop コマンドと同じように指定します。また、BuildLoop コマンドと同様、開始のアンカーを「C-N-CA」、終了のアンカーを「CA-C-N」の形式で指定すると、欠損部分が 17 残基以内であれば PDB のクイックアクセスインデックスを使用した即時検索が適用されます（メニューから実行する際も同様）。

③のオプションでは、PDB から抽出するサンプル数と二次構造情報を指定できます。詳細は、ユーザーマニュアル OptimizeLoop コマンドページをご参照ください。今回は、オプションを指定せずに実行します。（指定しない場合、サンプル数はデフォルトで 100 に設定されます。）

以下にコマンド例を示すので、先ほどモデリングした3つの欠損部分を最適化してみてください。この例ではアンカー原子には即時検索が適用される形式を使用しています。

```
OptimizeLoop C Res 230 or N CA Res 231,CA C Res 251 or N Res 252
OptimizeLoop C Res 304 or N CA Res 305,CA C Res 310 or N Res 311
OptimizeLoop C Res 368 or N CA Res 369,None
```

4.3.2 GUI メニューから実行する場合

メニューから実行する場合は、次の操作を行います。メニューから Edit > Optimize を選択し、ループの最適化を行いたい場合は Loop を、N 末端の場合は N-terminal loop、C 末端の場合は C-terminal loop をそれぞれ選択します。

1)　ループ（Res 232-250）の最適化

❑ メニューから、Edit > Optimize > Loop を選択

❑ 開始アンカー原子の選択

BuildLoop コマンドの実行時と同じように選択し、OK をクリック。（例：Res 230 の C と Res 231 の N、CA）

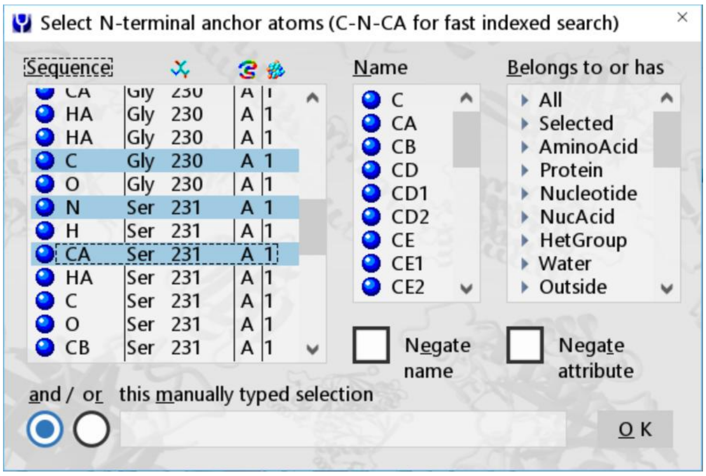

Figure 4-4　開始アンカー原子の選択例

- ❑ 終了アンカー原子の選択

こちらも BuildLoop コマンドの実行時と同じように選択し、OK をクリック。（例：Res 251 の CA、C と Res 252 の N）

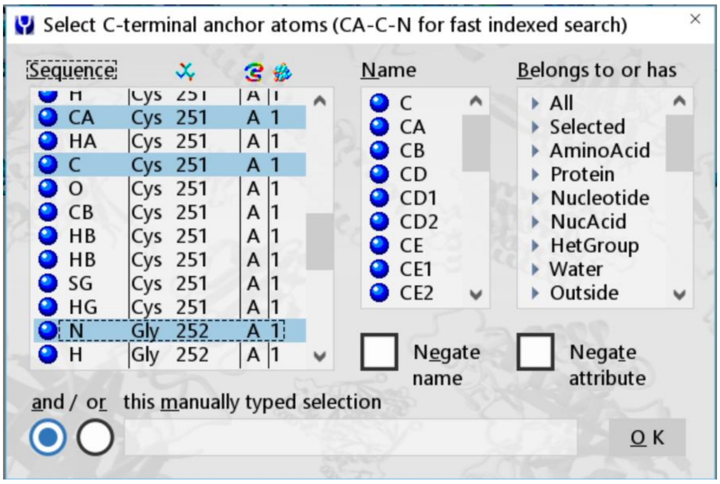

Figure 4-5　終了アンカー原子の選択例

- ❑ オプション設定・実行

　次に、オプションの設定画面が開きます。「Samples」では PDB から抽出する構造の数を指定できるので、任意に変更してください。コマンドから実行する場合のデフォルト値が 100 なので、ここでは例として 100 に設定し、OK をクリックして実行します。今回は空欄のままにしていますが、下の欄には二次構造の情報を指定することもできます。

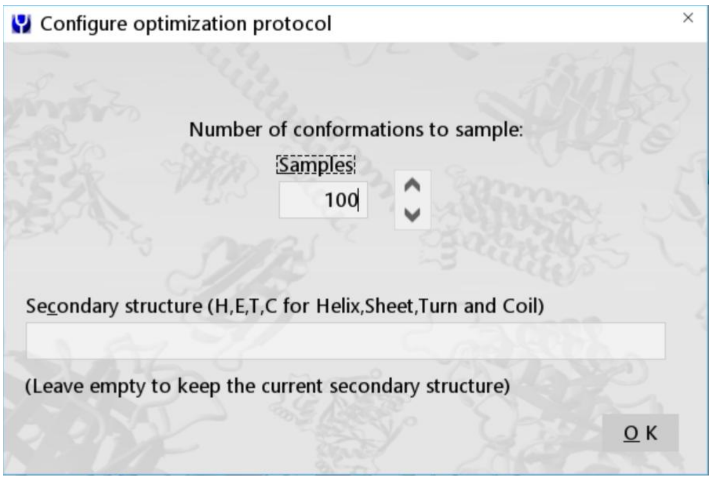

Figure 4-6　オプション設定画面

2) **ループ（Res 306-309）の最適化**

1)と同様に操作し、ループの最適化を行ってください。

- ❑ メニューから、Edit > Optimize > Loop を選択
- ❑ 開始アンカー原子の選択 （例：Res 304 の C と Res 305 の N、CA）
- ❑ 終了アンカー原子の選択 （例：Res 310 の CA、C と Res 311 の N）
- ❑ オプション設定・実行 （例：Samples 数を 100 に設定（任意））

3) **C 末端（Res 370-385）の最適化**

基本的には 1)と同じような操作になります。

- ❑ メニューから、Edit > Optimize > **C-terminal loop** を選択
- ❑ 開始アンカー原子の選択 （例：Res 368 の C と Res 369 の N、CA）
- ❑ オプション設定・実行 （例：Samples 数を 100 に設定（任意））

4.4 モデルの品質チェック（最適化後）

モデルの最適化が終了したら、再度 Z-スコアを確認してみます。

- ❑ Analyze > Check > All から、「ModelQuality:~」を選択し、OK

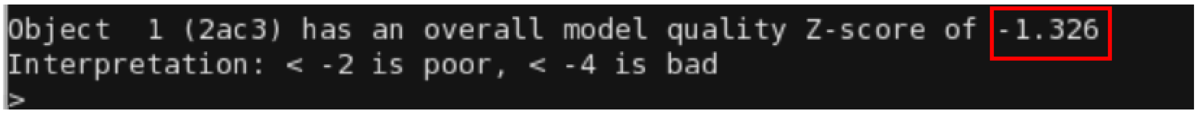

Figure 4-7 Z-スコアの確認②

最適化により、Z-スコアが -1.895 から -1.326 に改善することができました。

欠損したループや末端のモデリングの操作は以上となりますので、作成したモデルを File > Save as から保存して終了となります。Z-スコアをさらに改善したい場合は、そのまま次の手順「5 構造全体のリファイン」に進んでください。

5. 構造全体のリファイン

モデルのクオリティをさらに改善したい場合、主に次の 2 つの方法があります。1 つ目は、構造全体のエネルギー最小化計算を行う方法で、2 つ目は、構造リファイン用の短い MD 計算マクロ（md_refine.mcr）を利用する方法です。短い時間で実行したい場合は前者、時間をかけられる場合は後者を選択してください。

前者のエネルギー最小化については、弊社のウェブサイトに日本語チュートリアルを公開していますので、そちらを参考に実行してください。（YASARA 技術情報ページ https://www.affinity-science.com/yasara-tech/）

ここでは、後者の構造リファイン用のマクロファイルを実行する方法を紹介します。

5.1 構造リファイン用マクロファイルの実行

YASARA には構造リファイン用の短いシミュレーション実行マクロ、md_refine.mcr が付属しています。このマクロを実行すると、500 ps のシミュレーションが行われ、25 ps ごとに PDB ファイルが保存されます。さらに、各スナップショットのエネルギー値などを解析した結果ファイル（.tab ファイル）が作成されます。

- **作業ディレクトリの作成**

  マクロを実行するにあたり入出力ファイルを格納するディレクトリ（フォルダ）を新規に作成します（Windows ではエクスプローラ、Linux では mkdir コマンド等を使用）。作成場所やファイル名は任意で問題ありませんが、YASARA 上で表示が乱れるため、日本語を含まないようにしてください。

- **構造ファイルの保存**

  モデリングした構造ファイルを先ほど作成した作業ディレクトリに保存します。
  メニューの File > Save as > PDB file から、保存するオブジェクト（2ac3）を選択して OK、Browse から先ほど作成した作業ディレクトリを保存先に指定し、Filename 欄に任意のファイル名（例：2ac3-model.pdb など）を指定して OK。
  ※ファイル形式は、PDB ファイル形式の他に YASARA Object 形式ファイル（.yob）形式も利用できます。

- **マクロターゲットの指定**

  メニューの Options > Macro&Movie > Set target から、マクロターゲットを設定します。
  作業ディレクトリから先ほどで保存した構造ファイル（例：2ac3-model.pdb など）を選択し、右側の Remove…オプションの「file extension」にチェックを入れて拡張子を除外して指定します。

- **マクロファイルの実行**

  メニューの Options > Macro&Muvie > Play macro から、md_refine.mcr を選んで OK をクリックすると、MD 計算が開始します。

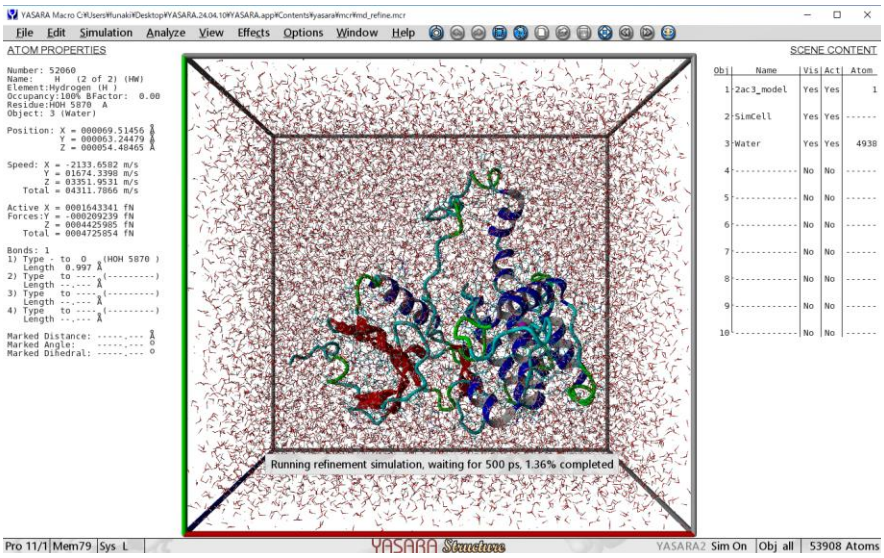

Figure 5-1 md_refine.mcr 実行中画面

❑ 結果ファイルの確認

シミュレーションが終了すると、マクロターゲットに指定した構造ファイルの保存先のフォルダに、

(MacroTarget) _results.tab （例：2ac3-model_results.tab）

という名前のタブ区切りテキストファイルが作成されます。このファイルを開くと、各スナップショットの解析結果を確認できます。最終行には、最もエネルギーが低い構造のスナップショットの情報が記載されています。同じフォルダに、各スナップショットの PDB ファイルも保存されます。今回の実行例では、スナップショット 18 のスコアが最もよかったので、今回はこの構造を採用することとします。(2ac3-model_snapshot18.pdb)

Snapshot	Energy	Dihedrals	Packing1D	Packing3D	Average
0.00	-159635.05	0.81	-0.96	-0.97	-0.37
1.00	-170604.56	1.26	-0.95	-0.82	-0.17
2.00	-169743.88	1.35	-0.92	-0.92	-0.16
3.00	-169498.28	1.31	-1.02	-0.95	-0.22
4.00	-170804.50	1.36	-0.80	-0.87	-0.10
5.00	-171401.96	1.46	-0.88	-0.75	-0.06
6.00	-173161.85	1.61	-0.81	-0.71	0.03
7.00	-172950.02	1.65	-0.87	-0.69	0.03
8.00	-171616.61	1.53	-0.87	-0.73	-0.02
9.00	-171674.77	1.56	-1.01	-0.74	-0.06
10.00	-172847.82	1.54	-0.87	-0.62	0.02
11.00	-172970.72	1.62	-0.93	-0.64	0.02
12.00	-173408.83	1.53	-0.90	-0.66	-0.01
13.00	-172696.69	1.51	-0.86	-0.74	-0.03
14.00	-173871.54	1.65	-0.93	-0.63	0.03
15.00	-172729.95	1.54	-0.91	-0.68	-0.02
16.00	-174044.84	1.50	-0.90	-0.62	-0.01
17.00	-173170.53	1.68	-0.97	-0.68	0.01
18.00	-174456.89	1.70	-0.91	-0.62	0.05
19.00	-174078.60	1.60	-0.85	-0.67	0.03

Snapshot 18 has minimum energy -174456.89 and snapshot 18 has maximum quality score 0.055

Figure 5-2　結果ファイルの出力例

❑ 終了後：モデルのクオリティ(Z-スコア)の確認

採用した構造について、再度 Check コマンドを実行し、Z-スコアを確認してみます。

一旦画面をクリア(Clear Scene アイコン( )をクリック)し、先ほど採用した構造ファイルを開きます(File > Load > PDB file またはファイルをドラッグアンドドロップ)。続いて、Analyze > Check > All から、「ModelQuality:~」を選択し、OK をクリック。

```
Object  1 (2ac3_model_s) has an overall model quality Z-score of -0.365
Interpretation: < -2 is poor, < -4 is bad
```

Figure 5-3　Z-スコアの確認③

コンソール画面を確認すると、Z-スコアは-0.365 となり、マクロファイル実行前の-1.326 から大幅に改善することができました。

6. 参考情報

6.1 SampleLoop コマンドを利用したモデリングについて

BuildLoop コマンドと似ているコマンドに、SampleLoop があります。BuildLoop は、欠損している部分のモデルを構築するコマンドですが、SampleLoop は、既に存在しているループ(や末端)について、異なるコンフォメーションを生成するコマンドです。SampleLoop は既存のループを分析するため、BuildLoop と異なり欠損配列の入力は不要で、OptimizeLoop のようにアンカー原子を指定するだけで実行できます。実は、手順 2.1 のように PDB ファイルの読み込み時に SeqRes オプションを Yes に指定しておくと、欠損残基の情報も含めて読み込まれるため、SampleLoop コマンドでも PDB ファイルの欠損部分をモデリングすることができます。ただし、BuildLoop と SampleLoop を同じ条件で実行しても、同じ構造が得られるわけではないようです。(3 種類の PDB 構造を用いて両者のコマンドを実行してテストしたところ、いずれも BuildLoop の方がわずかに高品質なモデルを生成しました。)欠損配列を指定する必要がない分、SampleLoop コマンドを利用した方が簡単にモデリングを実行できますが、その点に留意してご利用ください。コマンドの詳細については、YASARA のユーザーマニュアルをご参照ください。

なお、YASARA ユーザーマニュアルの、LoadPDB コマンドページには、SampleLoop コマンドを利用し、自動で PDB ファイルの欠損部分のモデリングと最適化(OptimizeLoop コマンド)を行うことができるマクロ例が記載されていますので、よろしければご参照ください。

6.2 BuildLoop コマンドのオプションについて

BuildLoop コマンドには、次のオプションがあります。(詳細については、YASARA ユーザーマニュアルのコマンドページもあわせてご参照ください。)

オプション	指定内容	デフォルト値
Structures	生成したいモデル数を指定します。	1
Mutate	All | None | MismatchAla | Ala から選択 検索された PDB 構造と指定した配列の間の一致・不一致の処理方法を指定します。	All
Bumpsum	バックボーンのバンプの距離の合計が許容される最大値(Å)を指定します。	1.0
SecStr	二次構造を指定します。略号:H(Helix), E(Strand), T(Turn), C(Coil)	指定なし
BridgeCys	Yes | No で指定 生成したループと周辺残基の間にシステインのジスルフィド(S-S)結合を形成するか否かを指定します。	Yes

(補足)

・Mutate オプション

入力配列と検索されたデータベース配列の間の一致・不一致の場合の処理方法について、以下の表にまとめます。BuildLoop コマンドでは、入力配列が不明な場合は「X」(未知のアミノ酸コード)を利用できます。一番右の列は、入力配列が X の場合の処理内容になります。

入力配列:BuildLoop コマンド実行時に入力した、欠損残基の配列

データベース配列:BuildLoop コマンド実行時に検索された PDB データベースの配列

Mutate	一致	不一致	不明(X)残基
All	入力配列	入力配列	データベース配列
None	データベース配列	データベース配列	データベース配列
Ala	アラニン(Ala)	アラニン(Ala)	アラニン(Ala)
MismatchAla	入力配列	アラニン(Ala)	データベース配列

・BridgeCys オプション

「Yes」に設定して実行し、生成したループと周辺残基との間にジスルフィド結合が形成された場合、その後 OptimizeLoop コマンドを実行するとジスルフィド結合が乱れる可能性があります。本チュートリアルの例ではジスルフィド結合が形成されないため、オプションを設定しませんでしたが、OptimizeLoop コマンドを実行する予定がある場合には、「No」に設定することが推奨されています。

・GUI のオプション設定画面

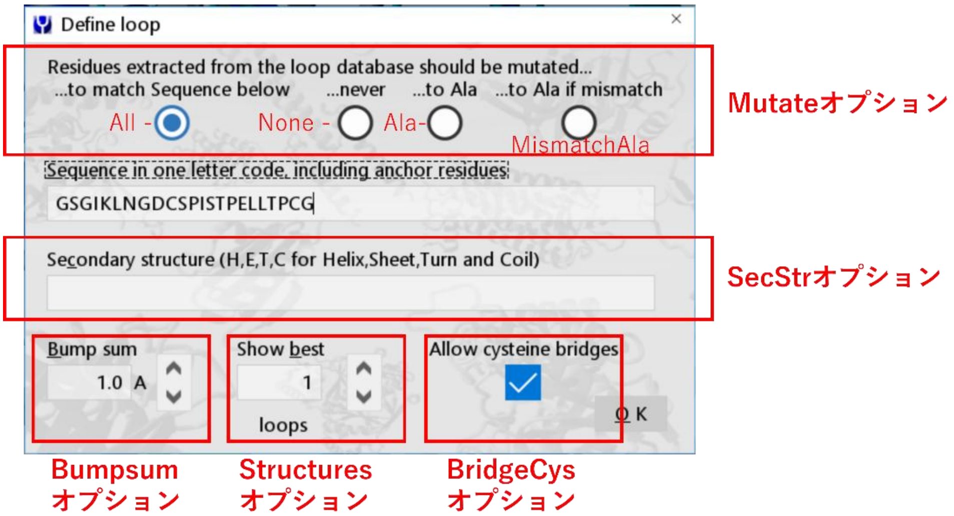

Figure 6-1　GUI のオプション設定画面

6.3 ユーザーマニュアルの関連ページについて

本チュートリアルで使用した各コマンドや操作の詳細については、YASARA ユーザーマニュアル（Help > Show user manual）の、以下の項目をご覧ください。

- 関連コマンドページ
  Commands - Tell YASARA what to do > Index - All commands in alphabetic order >
  ・BuildLoop - Build central or terminal loop
  ・OptimizeLoop - Optimize central or terminal loop
  ・SampleLoop - Sample central or terminal loop

- タンパク質構造のリファインについて
  Recipes - Perform complex tasks > Refine a protein model

6.4 その他の参考資料

エネルギー最小化や分子動力学計算など、弊社ウェブサイトにて各種チュートリアルを公開しています。その他技術情報については、ブログ記事や FAQ もあわせてご参照ください。

YASARA 日本語チュートリアル
https://www.affinity-science.com/yasara-tutorial

Affinity Science Blog/.org
https://www.affinity-science.org/

YASARA よくある質問
https://www.affinity-science.com/yasara-faq/

以上