Missing residues - Modeller Wiki

salilab.org/modeller/wiki/Missing residues

Modeller
Program for Comparative Protein Structure Modelling by Satisfaction of Spatial Restraints

Missing residues

Often, you will encounter files in the PDB which have missing residues. Special care must be taken in this case, as MODELLER only reads the ATOM and HETATM records, not the SEQRES records, and so will not handle missing residues automatically. (Unfortunately PDB is not reliable enough to be able to automatically rely on SEQRES.)

One example is PDB code 1qg8, which is missing residues 134-136 and 218-231 (see the REMARK 465 lines in the PDB file). We can use Modeller to 'fill in' these missing residues by treating the original structure (without the missing residues) as a template, and building a comparative model using the full sequence.

First, we obtain the sequence of residues with known structure:

Toggle line numbers

```
from modeller import *
# Get the sequence of the 1qg8 PDB file, and write to an alignment file
code = '1qg8'

e = Environ()
m = Model(e, file=code)
aln = Alignment(e)
aln.append_model(m, align_codes=code)
aln.write(file=code+'.seq')
```

This produces a sequence file, `1qg8.seq`:

```
>P1;1qg8
structureX:1qg8:   2 :A: 256 :A:undefined:undefined:-1.00:-1.00
PKVSVIMTSYNKSDYVAKSISSILSQTFSDFELFIMDDNSNEETLNVIRPFLNDNRVRFYQSDISGVKERTEKTR
YAALINQAIEMAEGEYITYATDDNIYMPDRLLKMVRELDTHPEKAVIYSASKTYHLNDIVKETVRPAAQVTWNAP
CAIDHCSVMHRYSVLEKVKEKFGSYWDESPAFYRIGDARFFWRVNHFYPFYPLDEELDLNYITEFVRNLPPQRNC
RELRESLKKLGMG*
```

From the PDB REMARKs or SEQRES records, we know the missing residues, so now we can make an alignment between the original 1qg8 structure (as the template), with gap characters corresponding to the missing residues, and the full sequence. This we place in a new alignment file, `alignment.ali`:

```
>P1;1qg8
structureX:1qg8:   2 :A: 256 :A:undefined:undefined:-1.00:-1.00
PKVSVIMTSYNKSDYVAKSISSILSQTFSDFELFIMDDNSNEETLNVIRPFLNDNRVRFYQSDISGVKERTEKTR
YAALINQAIEMAEGEYITYATDDNIYMPDRLLKMVRELDTHPEKAVIYSASKTYHLN---DIVKETVRPAAQVTW
NAPCAIDHCSVMHRYSVLEKVKEKFGSYWDESPAFYRIGDARFFWRVNHFYPFYPLDEELDLNYIT---------
-----EFVRNLPPQRNCRELRESLKKLGMG*
>P1;1qg8_fill
sequence::::::::
PKVSVIMTSYNKSDYVAKSISSILSQTFSDFELFIMDDNSNEETLNVIRPFLNDNRVRFYQSDISGVKERTEKTR
YAALINQAIEMAEGEYITYATDDNIYMPDRLLKMVRELDTHPEKAVIYSASKTYHLNENRDIVKETVRPAAQVTW
NAPCAIDHCSVMHRYSVLEKVKEKFGSYWDESPAFYRIGDARFFWRVNHFYPFYPLDEELDLNYITDQSIHFQLF
ELEKNEFVRNLPPQRNCRELRESLKKLGMG*
```

We can now use the standard Modeller 'LoopModel' class to generate a model with all residues, and then to refine the loop regions:

Toggle line numbers

```
from modeller import *
from modeller.automodel import *    # Load the AutoModel class

log.verbose()
env = Environ()

# directories for input atom files
env.io.atom_files_directory = ['.', '../atom_files']

a = LoopModel(env, alnfile = 'alignment.ali',
              knowns = '1qg8', sequence = '1qg8_fill')
a.starting_model= 1
a.ending_model  = 1

a.loop.starting_model = 1
a.loop.ending_model   = 2
a.loop.md_level       = refine.fast

a.make()
```

If you do not want loop refinement, simply use the `AutoModel` class rather than `LoopModel`, and remove the three lines which set `a.loop` parameters.

⚠ Note that loop modeling will only refine the shorter of the two loops by default. You can modify the `select_loop_atoms` routine to refine both loops, but you are not likely to get good results with this long insertion. In this case, you should probably try to find another template for this part of the sequence, or impose secondary structure restraints if you have reason to believe the insertion is not a loop.

Because either AutoModel or LoopModel will build a comparative model using your input PDB as a template, potentially all of the atoms in your final model could move. If you really don't want the non-missing residues to move, you can override the select_atoms method to select only the missing residues with a script similar to that below (note that the residue numbers are off by 1, since Modeller numbers the model starting at 1 in chain A, while the original PDB started numbering at 2):

Toggle line numbers

```
from modeller import *
from modeller.automodel import *    # Load the AutoModel class

log.verbose()
env = Environ()

# directories for input atom files
env.io.atom_files_directory = ['.', '../atom_files']

class MyModel(AutoModel):
    def select_atoms(self):
        return Selection(self.residue_range('133:A', '135:A'),
                         self.residue_range('217:A', '230:A'))

a = MyModel(env, alnfile = 'alignment.ali',
            knowns = '1qg8', sequence = '1qg8_fill')
a.starting_model= 1
a.ending_model  = 1

a.make()
```